

## Contents

3.2.3	Using the VAX/VMS Mail Utility	3-38
<hr/>		
3.3	LOGGING OFF THE SYSTEM	3-39
3.3.1	Logging Out with Video Terminals	3-42
3.3.2	Logging Out with Hardcopy Terminals	3-44
3.3.3	Logging Out with Disconnected Processes	3-44
3.3.4	Logging Out from a Dialup Login	3-45
<hr/>		
3.4	SUMMARY OF GOOD USER PRACTICES	3-45
<hr/>		
CHAPTER 4 FILE PROTECTION FEATURES		4-1
<hr/>		
4.1	HOW THE SYSTEM DETERMINES ACCESS	4-2
<hr/>		
4.2	STANDARD UIC-BASED PROTECTION	4-3
4.2.1	UICs and Protection	4-3
4.2.2	Specifying UICs	4-4
4.2.2.1	Numeric Format UICs	4-4
4.2.2.2	Alphanumeric Format UICs	4-5
4.2.2.3	UIC Translation and Storage	4-5
4.2.3	How UIC-based Protection Controls Access	4-6
4.2.4	Protection Code Syntax	4-8
4.2.5	How Privileges Affect Protection	4-9
4.2.6	How the System Interprets a Protection Code	4-10
4.2.7	How the System Interprets the Access Types for Objects	4-11
4.2.7.1	Access to Disk Files	4-11
4.2.7.2	Access to Directory Files	4-12
4.2.7.3	Access to Volumes	4-14
4.2.7.4	Access to Global Sections	4-14
4.2.7.5	Access to Devices	4-14
4.2.7.6	Access to Logical Name Tables	4-15
4.2.7.7	Access to Queues	4-15
4.2.8	Establishing and Changing UIC-Based Protection	4-16
4.2.8.1	Establishing and Changing Volume Protection	4-16.1
4.2.8.2	Establishing and Changing Directory Protection	4-16.1

## Contents

4.2.8.3	Establishing and Changing File Protection • 4-16.2	
4.2.8.4	Establishing and Changing Global Section Protection • 4-16.3	
4.2.8.5	Establishing and Changing Device Protection • 4-16.3	
4.2.8.6	Establishing and Changing Logical Name Table Protection • 4-16.3	
4.2.8.7	Establishing and Changing Queue Protection • 4-16.3	
<hr/>		
<b>4.3</b>	<b>ACCESS CONTROL LISTS (ACLS)</b>	<b>4-17</b>
4.3.1	ACLs, Identifiers, and the Reference Monitor	4-17
4.3.2	Creating and Maintaining ACLs	4-20
4.3.2.1	Object Types • 4-21	
4.3.3	Types of Identifiers	4-22
4.3.3.1	UIC Identifiers • 4-22.1	
4.3.3.2	General Identifiers • 4-22.1	
4.3.3.3	System-Defined Identifiers • 4-23	
4.3.3.4	Storage of Process Identifiers in the Rights List • 4-23	
4.3.4	Types of Access Control List Entries	4-24
4.3.4.1	Identifier ACE • 4-26	
4.3.4.2	Default Protection ACE • 4-30	
4.3.4.3	Security Alarm ACE • 4-31	
4.3.5	Managing Access Control Lists	4-34
<hr/>		
<b>4.4</b>	<b>ESTABLISHING AND CHANGING OWNERSHIP</b>	<b>4-35</b>
4.4.1	Understanding the Role of the Resource Attribute	4-35
4.4.2	Defining the Conditions that Convey Ownership Privileges	4-36
4.4.3	Establishing and Changing Volume Ownership	4-37
4.4.4	Establishing and Changing Directory Ownership	4-38
4.4.5	Establishing and Changing File Ownership	4-39
<hr/>		
<b>4.5</b>	<b>PROPAGATION OF PROTECTION DEFAULTS</b>	<b>4-40</b>
4.5.1	Default Directory File Protection	4-40
4.5.1.1	Default UIC-Based Directory File Protection • 4-40	
4.5.1.2	Default ACL Protection • 4-40	



## Contents

4.5.2	Default File Protection	4-41
4.5.2.1	Default UIC-based Protection • 4-41	
4.5.2.2	Default ACL Protection • 4-42	
4.5.2.3	File Protection Considerations • 4-43	
4.6	SUMMARY OF FILE PROTECTION EVALUATION	4-43
4.7	ESCAPING THE DISK SCAVENGER	4-48
4.8	MANAGING YOUR FILES FOR OPTIMUM SECURITY	4-50
4.9	AUDITING FOR THE USER	4-52
4.9.1	Noting Your Last Login Time	4-52
4.9.2	Tools for Detecting System Abuse	4-54
4.9.2.1	Security Alarms • 4-54	
4.9.2.2	Auditing Access to Sensitive Files • 4-55	
4.10	SUMMARY OF FILE PROTECTION TECHNIQUES	4-57
4.11	ANSWERS TO THE PROTECTION CHECKING SELF-TEST	4-57
CHAPTER 5 SECURITY FOR THE SYSTEM MANAGER		5-1
5.1	SECURITY MANAGEMENT ACCOUNT	5-2
5.2	CONSIDERATIONS FOR ESTABLISHING USER ACCOUNTS	5-2
5.2.1	Introduction to Group Design	5-3
5.2.2	Introduction to ACL Design	5-5
5.2.3	Introduction to Identifier Design	5-7
5.2.4	Some Special-Purpose Identifiers	5-9
5.2.5	Creating and Maintaining a Rights Database	5-9
5.2.5.1	Adding Identifiers • 5-10	
5.2.5.2	Adding Holders of Identifiers • 5-10	
5.2.5.3	Removing Identifiers and Holders • 5-11	
5.2.5.4	Displaying the Rights Database • 5-13	

## Contents

<b>5.2.6</b>	<b>Setting Protection and Ownership Defaults for Users</b>	<b>5-14</b>
5.2.6.1	Adjusting Protection Defaults • 5-18	
5.2.6.2	Adjusting Ownership • 5-20	
<b>5.2.7</b>	<b>Password Management</b>	<b>5-23</b>
5.2.7.1	Initial Passwords • 5-24	
5.2.7.2	Password Verification • 5-25	
5.2.7.3	System Passwords • 5-26	
5.2.7.4	Primary and Secondary Passwords • 5-28	
5.2.7.5	Enforcing Minimum Password Standards • 5-29	
5.2.7.6	Requiring the Password Generator • 5-32	
5.2.7.7	Protecting Passwords • 5-33	
<b>5.2.8</b>	<b>Login Options</b>	<b>5-35</b>
5.2.8.1	Controlling the Announcement Message • 5-35	
5.2.8.2	Controlling Disconnected Jobs • 5-35	
5.2.8.3	Controlling the Welcome Message • 5-36	
5.2.8.4	Controlling the Last Login Messages • 5-36	
5.2.8.5	Controlling New Mail Announcements • 5-37	
5.2.8.6	Controlling the Number of Retries on Dialups • 5-37	
5.2.8.7	Controlling Breakin Detection and Evasion • 5-38	
5.2.8.8	Using the Secure Server • 5-42	
<b>5.2.9</b>	<b>Using the Automatic Login Facility</b>	<b>5-43</b>
5.2.9.1	Adding New Records • 5-43	
5.2.9.2	Modifying Records • 5-44	
5.2.9.3	Deleting Records • 5-44	
5.2.9.4	Exiting from ALFMAINT • 5-44	
5.2.9.5	Logging in on an Automatic Login Terminal • 5-45	
5.2.9.6	Protecting Automatic Login Accounts • 5-46	
<hr/>		
<b>5.3</b>	<b>AUTHORIZING USAGE</b>	<b>5-46</b>
<b>5.3.1</b>	<b>Restricting Devices</b>	<b>5-46</b>
5.3.1.1	Restricting Terminal Usage • 5-47	
5.3.1.2	Restricting Disk Volumes • 5-47	
5.3.1.3	Applications Terminals and Miscellaneous Devices • 5-47	
<b>5.3.2</b>	<b>Restricting Work Times</b>	<b>5-48</b>
<b>5.3.3</b>	<b>Restricting Mode of Operation</b>	<b>5-49</b>
<b>5.3.4</b>	<b>Restricting Command Usage</b>	<b>5-50</b>
<b>5.3.5</b>	<b>Restricting Account Duration</b>	<b>5-50</b>

## Contents

5.3.6	Granting User Privileges	5-51
5.3.6.1	Alternatives to Privilege • 5-54	
5.3.6.2	Suggested Privilege Allocations • 5-57	
5.3.6.3	Controlling Privileged Accounts • 5-58	
5.3.6.4	Special-purpose Privileged Captive Accounts • 5-59	
5.3.7	Examples of Establishing User Accounts	5-59
5.3.7.1	A System Manager's Account • 5-60	
5.3.7.2	A Typical Interactive User's Account • 5-61	
5.3.7.3	A Production Account • 5-61	
5.3.8	Educating the New User	5-62
5.4	PROTECTING INFORMATION	5-68
5.4.1	Restricting Command Outputs	5-70
5.4.2	Protecting System Programs and Databases	5-70
5.4.3	Precautions to Take when Installing New Software	5-71
5.4.3.1	Avoiding Trojan Horse Programs • 5-71	
5.4.3.2	Installing Programs with Privilege • 5-72	
5.4.3.3	Protecting Programs and Directories • 5-73	
5.5	DISK MAINTENANCE CONSIDERATIONS	5-74
5.6	METHODS FOR DISCOURAGING DISK SCAVENGING	5-75
5.6.1	Erasing Techniques	5-75
5.6.2	Prevention Through Highwater Marking	5-77
5.6.3	Summary of Prevention Techniques	5-78
5.7	RESTRICTED ENVIRONMENTS	5-78
5.7.1	Creating a Captive Account	5-79
5.7.1.1	Login Command File Considerations • 5-81	
5.7.1.2	Guest Accounts • 5-83	
5.7.1.3	Proxy Login Accounts • 5-86	
5.8	AUDITING WITH SECURITY ALARMS	5-86
5.8.1	Alarm Events	5-86
5.8.2	Enabling a Security Operator Terminal	5-86.1
5.8.3	Alarm Messages	5-86.2
5.8.4	Audit Reduction Facility	5-86.2
5.8.4.1	Optional Parameters • 5-86.3	
5.8.5	Auditing a Terminal Session	5-86.4
5.8.6	Enforcing a Terminal Session Audit	5-86.5

## Contents

5.8.7	Other Audit Data	5-86.7
5.9	ONGOING TASKS	5-86.7
5.9.1	General Surveillance	5-87
<b>CHAPTER 6 WHEN YOUR SYSTEM IS UNDER ATTACK</b>		<b>6-1</b>
6.1	INDICATIONS OF TROUBLE	6-2
6.1.1	User Indications	6-2
6.1.2	Personal Observations	6-4
6.2	ROUTINE SYSTEM SURVEILLANCE	6-5
6.2.1	Accounting Log	6-5
6.2.2	Security Auditing	6-6
6.3	HANDLING A SECURITY BREACH	6-7
6.3.1	Unsuccessful Breakin Attempts	6-8
6.3.1.1	Detection of the Unsuccessful Breakin Attempt • 6-8	
6.3.1.2	Identifying the Perpetrator • 6-8	
6.3.1.3	Prevention of Breakin Attempts • 6-9	
6.3.1.4	Repair After an Unsuccessful Breakin • 6-10	
6.3.2	Successful Breakin Attempts	6-10
6.3.2.1	Detection of Successful Breakin Attempts • 6-10	
6.3.2.2	Identification of Breakin Perpetrator • 6-10	
6.3.2.3	Prevention of Breakin Attempts • 6-12	
6.3.2.4	Repair After a Breakin • 6-13	
6.4	SUMMARY	6-13

<b>CHAPTER 7</b>	<b>SECURITY FOR A DECNET NODE</b>	<b>7-1</b>
<b>7.1</b>	<b>THE REFERENCE MONITOR IN A NETWORK</b>	<b>7-2</b>
7.1.1	Establishing Subject Correspondence	7-5
7.1.2	Specifying Authorizations	7-6
7.1.3	Protecting Communications	7-6
7.1.4	Summary of VAX/VMS Network Security and the Reference Monitor	7-7
<b>7.2</b>	<b>DECNET-VAX ACCOUNTS</b>	<b>7-7</b>
<b>7.3</b>	<b>THE DECNET-VAX DATABASE</b>	<b>7-9</b>
<b>7.4</b>	<b>NETWORK USAGE</b>	<b>7-10</b>
<b>7.5</b>	<b>SPECIFYING DECNET OBJECT ACCOUNTS</b>	<b>7-10</b>
<b>7.6</b>	<b>PROXY LOGINS</b>	<b>7-14</b>
7.6.1	Setting Up Proxy Logins	7-14
7.6.1.1	Using the VAX/VMS Authorize Utility • 7-15	
7.6.1.2	A Proxy Account • 7-15	
7.6.1.3	Using the VAX/VMS Network Control Program Utility • 7-17	
7.6.1.4	Conditions for Proxy Access • 7-19	
7.6.2	Special Considerations	7-20
<b>7.7</b>	<b>SHARING AND EXCHANGING FILES IN THE NETWORK ENVIRONMENT</b>	<b>7-20</b>
7.7.1	Many Remote Users Seek Access for a Single Purpose	7-21
7.7.2	Remote Users from One Node Require Single Account Access	7-22
7.7.3	A Few Outside Users Require Access for Multiple Purposes	7-23

---

**CHAPTER 8 SECURITY CONCERNS ON A CLUSTER 8-1**

---

<b>8.1</b>	<b>OVERVIEW OF CLUSTERS AND SECURITY CONSIDERATIONS</b>	<b>8-1</b>
<b>8.2</b>	<b>AUTHORIZATION DATABASE CONSIDERATIONS</b>	<b>8-2</b>
<b>8.3</b>	<b>BUILDING A COMMON USER ENVIRONMENT</b>	<b>8-3</b>
<b>8.4</b>	<b>FILE SHARING CONSIDERATIONS</b>	<b>8-3</b>
<b>8.5</b>	<b>USING DECNET BETWEEN CLUSTER NODES</b>	<b>8-4</b>
<b>8.6</b>	<b>SUMMARY</b>	<b>8-4</b>

---

**APPENDIX A PRIVILEGES A-1**

---

<b>A.1</b>	<b>USER PRIVILEGES</b>	<b>A-1</b>
A.1.1	ACNT Privilege _____	A-1
A.1.2	ALLSPOOL Privilege _____	A-1
A.1.3	ALTPRI Privilege _____	A-2
A.1.4	BUGCHK Privilege _____	A-2
A.1.5	BYPASS Privilege _____	A-3
A.1.6	CMEXEC Privilege _____	A-3
A.1.7	CMKRNL Privilege _____	A-4
A.1.8	DETACH Privilege _____	A-4
A.1.9	DIAGNOSE Privilege _____	A-4
A.1.10	EXQUOTA Privilege _____	A-5
A.1.11	GROUP Privilege _____	A-5
A.1.12	GRPNAM Privilege _____	A-5
A.1.13	GRPPRV Privilege _____	A-6
A.1.14	LOG_IO Privilege _____	A-6
A.1.15	MOUNT Privilege _____	A-7
A.1.16	NETMBX Privilege _____	A-7
A.1.17	OPER Privilege _____	A-7
A.1.18	PFNMAP Privilege _____	A-8
A.1.19	PHY_IO Privilege _____	A-8

## Contents

A.1.20 PRMCEB Privilege	A-9
A.1.21 PRMGBL Privilege	A-9
A.1.22 PRMMBX Privilege	A-10
A.1.23 PSWAPM Privilege	A-10
A.1.24 READALL Privilege	A-11
A.1.25 SECURITY Privilege	A-11
A.1.26 SETPRV Privilege	A-11
A.1.27 SHARE Privilege	A-12
A.1.28 SHMEM Privilege	A-12
A.1.29 SYSGBL Privilege	A-12
A.1.30 SYSLCK Privilege	A-12
A.1.31 SYSNAM Privilege	A-13
A.1.32 SYSPRV Privilege	A-13
A.1.33 TMPMBX Privilege	A-14
A.1.34 VOLPRO Privilege	A-14
A.1.35 WORLD Privilege	A-15

---

APPENDIX B USING THE USER DATA AREAS IN UAF RECORDS	B-1
---	-----

---

APPENDIX C PROTECTION FOR VAX/VMS SYSTEM FILES	C-1
--	-----

---

APPENDIX D RUNNING VAX/VMS IN A C2 ENVIRONMENT	D-1
--	-----

---

APPENDIX E ALARM MESSAGES	E-1
---------------------------	-----

---

E.1 ALARMS AUDITING ACCESS TO FILES AND GLOBAL SECTIONS	E-1
E.2 ALARMS REQUESTED BY AN ACL	E-2
E.3 ALARMS AUDITING INSTALL OPERATIONS	E-3
E.4 ALARMS RESULTING FROM MODIFICATIONS TO THE RIGHTS DATABASE	E-4

## Contents

E.5	ALARMS RESULTING FROM CHANGES TO SYSTEM OR NETWORK UAF	E-7
E.6	ALARMS RESULTING FROM PASSWORD CHANGES	E-9
E.7	BREAKIN ATTEMPT ALARMS	E-9
E.8	LOGIN ALARMS	E-11
E.9	LOGIN FAILURE ALARMS	E-12
E.10	LOGOUT ALARMS	E-14
E.11	VOLUME MOUNT AND DISMOUNT ALARMS	E-15
E.12	ALARMS RESULTING FROM EXECUTION OF SET AUDIT COMMAND	E-16

## GLOSSARY

Glossary-1

## INDEX

## FIGURES

2-1	Reference Monitor Diagram	2-2
3-1	Example of Local Login, Illustrating Messages	3-6
3-2	Illustration of File Sharing Over a Network	3-36
4-1	Illustrating User Categories with a UIC of [100,100]	4-7
4-2	Example of an Access Matrix	4-18
4-3	Previous Matrix with Labeled Crosspoints	4-19
4-4	Flowchart of Access Request Evaluation	4-45
5-1	Employee Grouping by Department and Function	5-4
5-2	Access Requirements for Payroll Application	5-6
5-3	Access Requirements for Order Processing	5-6

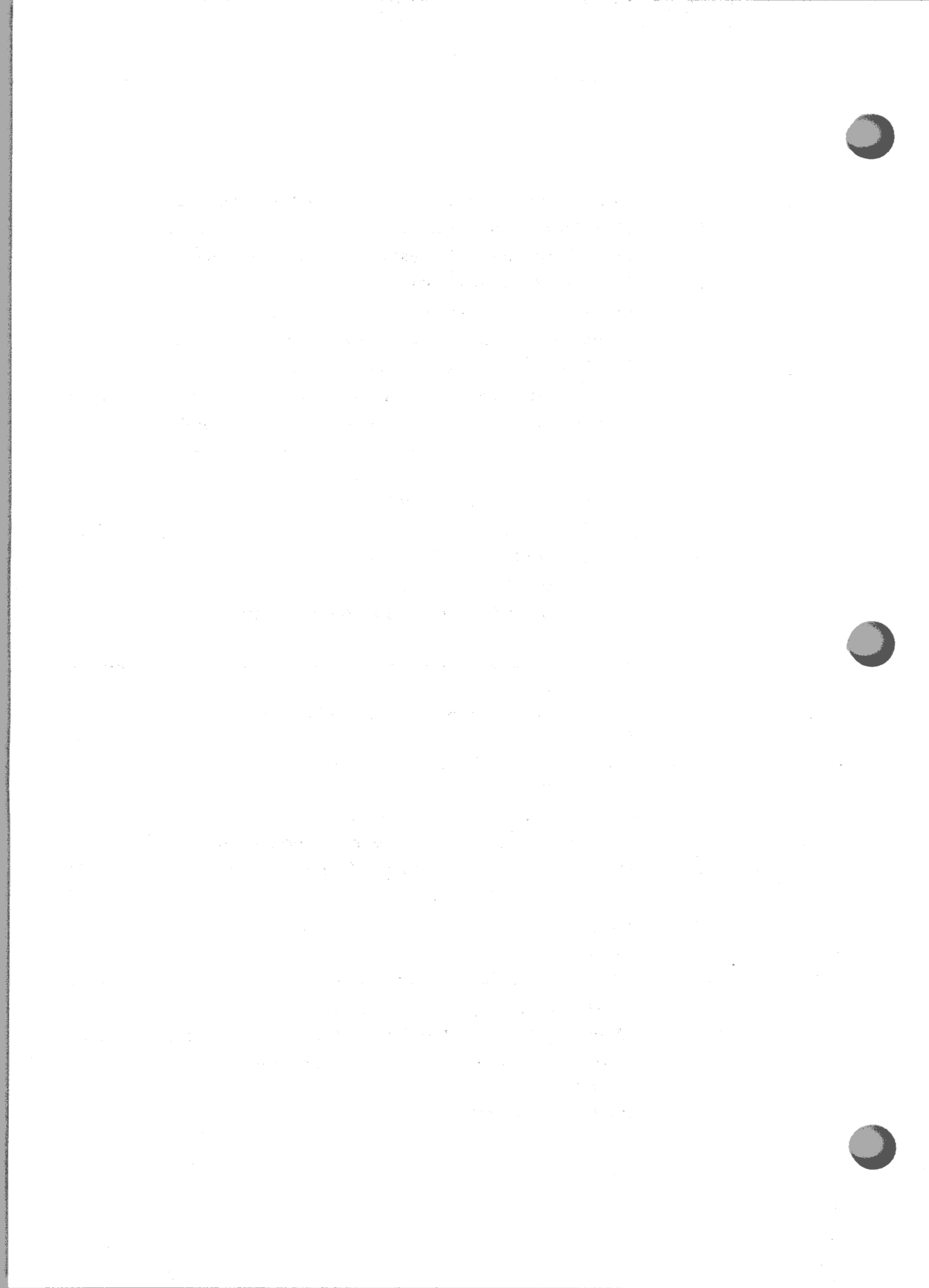


## Contents

5-4	Access Requirements for Accounts Receivable	5-7
5-5	Flowchart of File Creation	5-16
5-6	Example of a Security/System Manager's Account	5-60
5-7	Example of a Typical Interactive User Account	5-61
5-8	Example of a Production Account	5-62
5-9	Sample Form Introducing a New Account	5-66
5-10	Example of a Captive Command Procedure	5-84
5-11	Sample Captive Procedure for Privileged Accounts	5-85
7-1	Simple Diagram of Reference Monitor in a Network	7-3
7-2	Advanced Diagram of the Reference Monitor in a Network	7-4
7-1	Definitions in the Network Object Database	7-11
7-2	UAF Record for FAL Account	7-12
7-3	UAF Record for DECNET Account	7-13
7-4	Example of a Proxy Account	7-17
7-5	Example of Protected File Sharing in a Network	7-24

## TABLES

1-1	Target Sites and Typical Characteristics of Perpetrators	1-5
1-2	Event Tolerance As A Measure of Security Requirements	1-8
3-1	Classes and Types of Logins	3-2
3-2	Common Symptoms and Causes of Login Failures	3-30
3-3	Other Symptoms and Causes of Login Failures	3-34
4-1	Identifier ACE Options	4-27
4-2	Identifier ACE Access Types	4-28
4-3	Security Alarm ACE Options	4-32
4-4	Security Alarm ACE Access Types	4-33
5-5	VAX/VMS Privileges	5-53
5-6	Minimum Privileges For System Users	5-58
6-1	System File Candidates for ACL-based File Access Auditing	6-7
7-1	Access Control Parameter Values	7-18



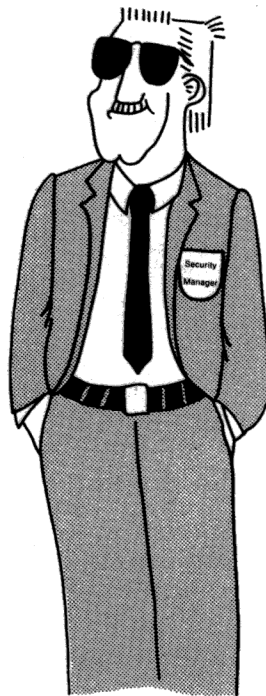
---

## Preface

---

### Intended Audience

This guide is designed for VAX/VMS users who assume overall responsibility for protecting their systems from myriad forms of tampering, observation, or theft of services by unauthorized users. Each site determines whether this job is suitable for a single individual or a group of individuals. This guide uses the title of *security manager* to refer to all such individuals. It is quite possible that at many sites the system manager assumes these duties, since many of the skills and responsibilities do overlap. However, it is also possible that many other sites will find that the security manager role requires a separate job designation, as well as specific training and privileges.



ZK-2055-84

## **Preface**

In this guide, narrative introduces each topic and explains why it is important. References or explanations provide the specifics. Generally, material presented elsewhere will not be duplicated unless deemed unusually beneficial. This guide is intended to be application-oriented, helpful, and friendly. Extensive examples are provided to clarify related operations.

---

## Structure of This Document

Chapters 1, 2, 3, and 4 are the only chapters written for the general user. They will be of interest as well to the security manager, but are written at the level of the general user who has acquired a knowledge of VAX/VMS up through and including acquaintance with the *VAX/VMS DCL Dictionary*. It seems likely that many security managers will distribute just Chapter 3 to their users expected to benefit from reading it. At sites where users are actively involved in file protection, Chapter 4 or excerpts from it might also be distributed.

The chapters are summarized as follows:

- Chapter 1 introduces the topics of security and levels of security requirements, explains the three sources of security failures, and sets expectations for the contents of this guide.
- Chapter 2 introduces the reference monitor concept for security design and provides an overview of the security features provided by VAX/VMS.
- Chapter 3 provides information on system security of interest to all users, not just security managers.
- Chapter 4 describes the VAX/VMS file protection features in detail. It is of interest to many users, as well as extremely valuable to all security managers.
- Chapter 5 describes the general system security features of particular interest to security managers.
- Chapter 6 considers how to recognize when a system is under attack and summarizes the actions available to security managers.
- Chapter 7 describes security considerations for systems using networking.
- Chapter 8 describes security-related actions specific to clustered systems, such as mounting the disks and setting up the authorization database.
- Appendix A summarizes all the privileges available on VAX/VMS systems, what they provide, and who may need them.

## Preface

- Appendix B describes how to use the user data areas in the User Authorization File.
- Appendix C lists the UIC-based protection that DIGITAL provides, by default, for system files.
- Appendix D describes how to operate VAX/VMS systems in a C2 environment.
- Appendix E lists the alarm messages resulting from particular alarm events.
- The Glossary offers definitions of the security related terms introduced in this guide.

## PHILOSOPHY

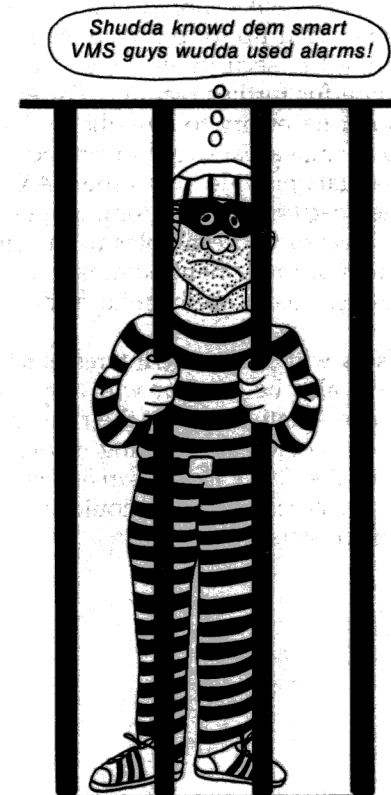
It is always possible for a security guide to become a gold mine of ideas when placed in the wrong hands. While this guide certainly does not intend to open vistas of exploration for malicious users by suggesting ways to thwart system security, it does not attempt to hide facts about vulnerabilities. The philosophy throughout this guide is that the security manager must understand the ramifications of various security exposures to appreciate why and when a security measure is appropriate.



24-2008-04

## Preface

As a result of the decision to openly discuss security weaknesses wherever they might exist, this guide divulges some common breakin tactics. However, when a potential breakin method is disclosed, one or more counteractive suggestions are discussed. The net effect should be to discourage many potential perpetrators. Unfortunately, there is no doubt that as a result of this guide the system that lies exposed to penetrations becomes further exposed. It becomes each site's responsibility to improve its own security by incorporating the suggestions herein.



ZK-2057-84

---

## Associated Documents

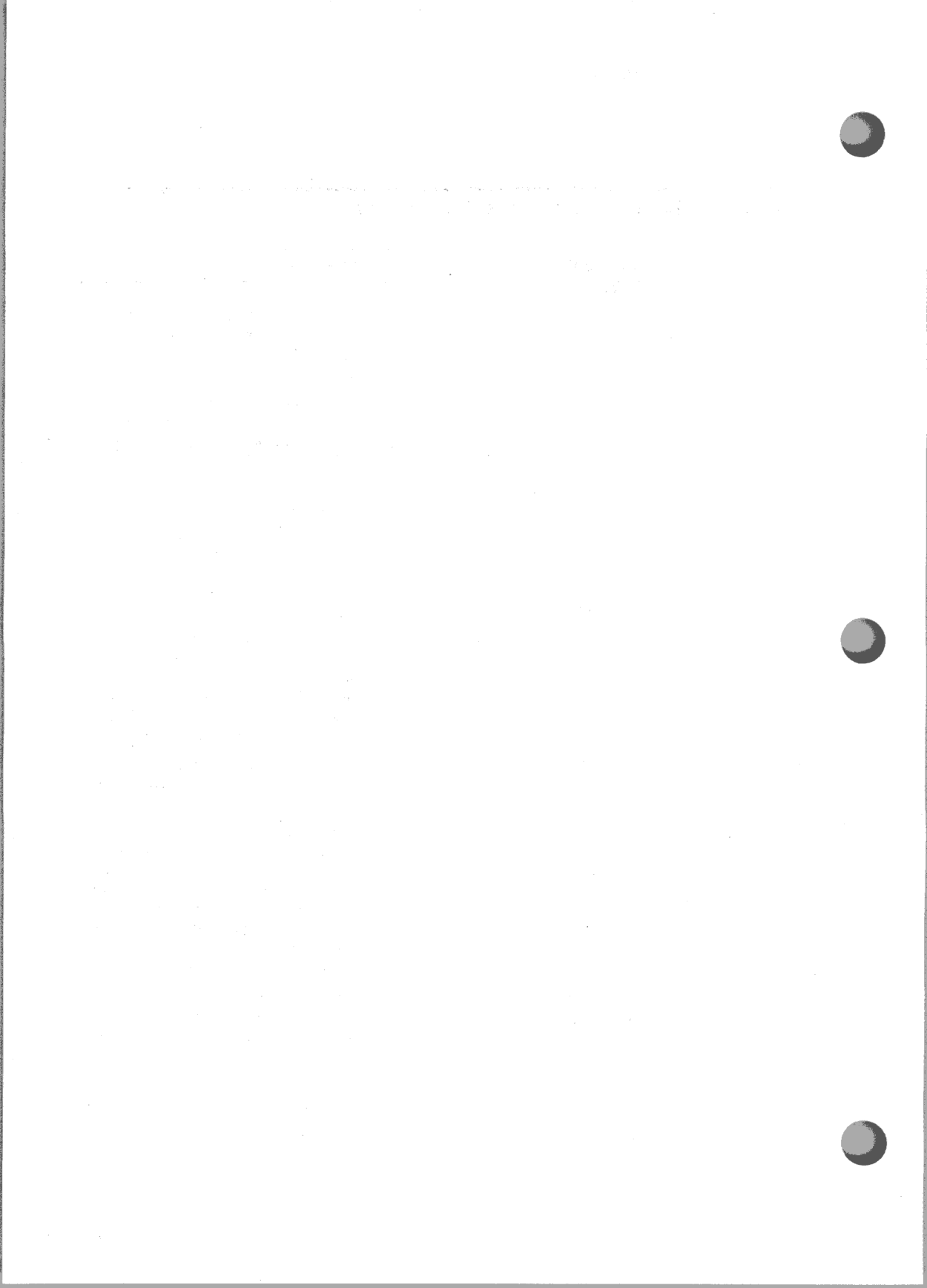
As a prerequisite, this guide assumes that the security manager has obtained the skills and training of a system manager. Thus, the *Guide to VAX/VMS System Management and Daily Operations* and all its prerequisites are mandatory reading, as a minimum. The *Guide to VAX/VMS System Security* focuses on topics of special interest to system security, presuming the knowledge level that the general reader would have gained through reading the *VAX/VMS DCL Dictionary* or *Guide to VAX/VMS System Management and Daily Operations*. Readers of those books should find their discussions of security-related topics lead up to, but stop short of, the advanced topics of interest to security managers. Readers are then directed to this guide for further reading. For example, this guide assumes it is not necessary to describe in detail simple steps such as how to change a password or protect or share a file (since reference manuals provided with the VAX/VMS documentation present the specifics for such single operations). However, expect instead to find examples in this guide that suggest how and when to implement automatic password generation, as well as discussions of the benefits and limitations of this technique.

Users with a specific interest such as networking or clusters, must also be familiar with the documents provided in the documentation set for their areas. That is, security managers in VAX/VMS networking environments should be familiar with the *Guide to Networking on VAX/VMS*. Security managers on clustered systems should be familiar with the *Guide to VAXclusters*.



## Conventions Used in This Document

Convention	Meaning
<code>RET</code>	A symbol with a one- to six-character abbreviation indicates that you press a key on the terminal, for example, <code>RET</code> .
<code>CTRL/x</code>	The phrase CTRL/x indicates that you must press the key labeled CTRL while you simultaneously press another key, for example, CTRL/C, CTRL/Y, CTRL/O.
\$ SHOW TIME 05-JUN-1985 11:55:22	Command examples show all output lines or prompting characters that the system prints or displays in black letters. All user-entered commands are shown in red letters.
\$ TYPE MYFILE.DAT . .	Vertical series of periods, or ellipsis, mean either that not all the data that the system would display in response to the particular command is shown or that not all the data a user would enter is shown.
file-spec,...	Horizontal ellipsis indicates that additional parameters, values, or information can be entered.
[logical-name]	Square brackets indicate that the enclosed item is optional. (Square brackets are not, however, optional in the syntax of a directory name in a file specification or in the syntax of a substring specification in an assignment statement.)
quotation marks apostrophes	The term quotation marks is used to refer to double quotation marks ("). The term apostrophe (') is used to refer to a single quotation mark.



---

## Summary of Technical Changes

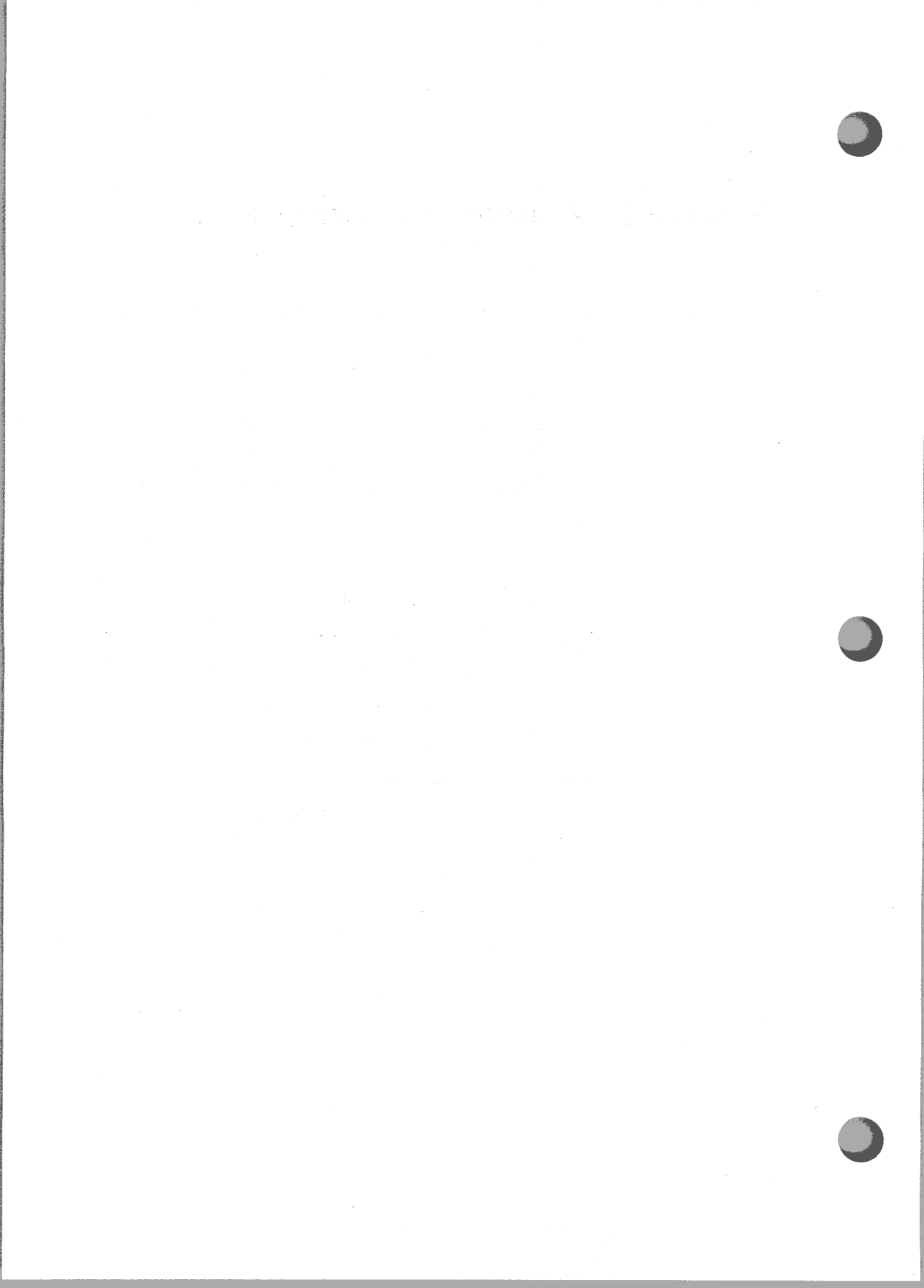
Version 4.2 of VAX/VMS contains a number of new security features:

- Access control lists (ACLs) on global sections and logical name tables
- Two DCL commands to access and maintain the breakin database

SHOW INTRUSION	Displays the contents of the breakin database
----------------	---

DELETE/INTRUSION	Removes an entry from the breakin database
------------------	--

- New alarms that audit the following events:
  - Any Install Utility operations
  - Any execution of the SET AUDIT command
  - Any system or user password changes
  - Modification of the rights database
  - Access to global sections
- Audit Reduction Facility—Using SECAUDIT.COM, you can selectively extract alarm information from the security operator's log.



# 1

---

## Introduction

The concept of security for a computer system differs little from the concept of security for your own home. When you think of securing your home, you want to keep unwelcome guests out so that they cannot snoop through your belongings, steal your possessions, consume your resources such as food, heat, water or telephone time, or damage your house or its contents. When you address security concerns for a computer system, you want to prevent outsiders from browsing through proprietary software or messages to gain valuable ideas or other confidential information. You also want to prevent outsiders from stealing copies of software or plans and prevent them from stealing services such as computer time. Finally, you want to protect the equipment, software, and files from damage.

However, the methods used to gain security differ considerably for the homeowner and computer site. Securing a computer site is a much more complex problem. While the homeowner and computer site share worries about break-ins from weak external physical security, the computer site must also be secured from damage by those deliberately admitted inside the site and those who may have or gain access from remote locations. You begin to see how complex the problem is when you recognize that computer sites frequently want to permit certain users certain kinds of restricted access to certain areas and not to others. This guide describes the problems computer sites face and the methods available to VAX/VMS sites to secure their VAX computers through software features.

This chapter provides an overview of the areas that must be carefully addressed by any site that desires a secure environment for computer operation. Some of these areas are the natural concerns of subsequent chapters of this guide, while other equally important areas are not appropriate for further discussion in this guide. This chapter should clarify the differences and set your expectations for the rest of the guide.

---

### 1.1 Types of Computer Security Problems

When you analyze the source of a *security breach* (or break) on any computer system, you typically find the source falls into one of three categories: user irresponsibility, user probing, or user penetration.

---

#### 1.1.1 User Irresponsibility

*User irresponsibility* refers to situations where the user does the wrong thing and causes some noticeable damage. Perhaps the user who is properly authorized to access certain files makes a copy of a key file and then sells the copy.

Unfortunately, there is little that an operating system such as VAX/VMS can do to protect sites from this source of security failures. The problem typically lies in application design deficiencies, or sloppy or inconsistent use of available controls by users and/or the security manager. Sometimes the failure to enforce adequate environmental security unwittingly encourages this type of security problem. None of these concerns is the proper domain of this guide.

Concerning the user irresponsibility problem, this guide will go only so far as to describe how you can restrict each user to a specific domain of activity and responsibility. It will also suggest ways to audit authorized actions. In this way, you may be able to reduce the potential harm from careless users and also identify who they are more swiftly. You can even hope that the aspect of human nature that responds to the fear of being singled out will work in your favor. However, if you fail to implement the suggestions, implement them inconsistently or poorly, or otherwise fail to motivate your users to observe good security practices, you will remain vulnerable to security failures caused by user irresponsibility.

---

### 1.1.2 User Probing

*User probing* refers to situations where a user exploits insufficiently protected parts of the system that result from inadequate use of existing security features. The user who succeeds in probing is usually somewhat knowledgeable, and is sometimes driven by intellectual challenge. These activities often start as "larks" where users just want to test their prowess against the system to possibly gain some forbidden access or machine time. Unfortunately, these perpetrators may get carried away by their successes, and theft of services is a crime. At any rate, the severity of the malicious intent can run the gamut from theft of services or invasion of confidential information, to embezzlement or destruction of data. Never lightly dismiss user probing.

VAX/VMS provides many security features that can be implemented to combat user probing. The features are applied according to the current needs at the site as judged by the security manager. Some of the features are best implemented on a temporary basis, while some are needed permanently. Some features include many options for your selection, while others you apply in an all-or-nothing manner. Some features you present to users as suggestions they should adopt, while other features you impose on the users, offering them no other recourse.

This guide helps you learn what these features are and how to use them appropriately. By studying this guide and skillfully applying and adhering to the techniques suggested, you can significantly reduce the number of security breaks caused by user probing.

Nevertheless, there remain areas of vulnerability within the operating system that are still not addressed. The very skilled malicious user may succeed in locating these weaknesses or may outsmart you by locating weaknesses in your particular application of the security techniques. This, in fact, is what the third category, penetration, is all about.

---

### 1.1.3 User Penetration

*Penetration* refers to situations where the user succeeds in breaking right through the controls. The controls are simply insufficient. While considerable effort has been expended to make this operating system resistant to penetration, some likelihood exists that penetration is still possible. There is little doubt that the skilled user who succeeds in penetration is malicious. This perpetrator is definitely not an amateur. Thus, this is the most serious and potentially dangerous type of security breach. However, in a well-run operation, it is also the rarest form, because of the skill levels and perseverance it requires from the perpetrator.

---

### 1.1.4 Characterizing the Perpetrators

You gain some perspective on the security problem when you analyze what kinds of sites are likely targets, what motivates the perpetrator, and what kinds of resources the perpetrator applies to the problem. It is just as important to understand the value the perpetrator sees in breaking security as it is to understand the consequence of the security break. To some extent, this value determines who the perpetrator will be and what resources will be applied.

Table 1-1 attempts to summarize these characteristics for four typical targets. Notice that all sites can suffer from user irresponsibility, but universities are common targets for user probing. Military and national security sites must particularly concern themselves about penetration.



**Table 1-1 Target Sites and Typical Characteristics of Perpetrators**

Target Site	Type of Perpetrator	Motivating Value to Perpetrator	Resources Applied
University	Student Hacker	Mostly fun, challenge, free software, free machine time	Time, cleverness, minimal money
Bank	White collar criminal	Money, revenge	Expertise, some money
Industry	White collar criminal	Trade secrets, revenge	Expertise, some money
Military	Opposing Power	World domination	Expertise, heavy financial investment top technology

The student hacker is most likely to be guilty of probing and, occasionally, penetration. White collar criminals typically represent cases of user irresponsibility and possibly probing. The outsider who gains access to a military or national security site will almost always represent a case of penetration.

## 1.2 The Secure System Environment

The description of the category of user irresponsibility briefly mentioned that environmental factors affect system security, but that they are not topics appropriate for this guide. Some clarification is due.

Environmental factors refer to dual sources of security problems outside the operating system domain—the personnel and the facilities. Security managers must do everything they can to motivate personnel to give high priority to security concerns. Security managers must also strive to keep the facility physically secure from intrusion by unwanted outsiders as well as from intrusive access by insiders.

## Introduction

To address the personnel issue, sites should have an explicit security program that motivates people to protect information and also limits and controls the exposure of sensitive information. The facilities aspect of the security problem demands that sites provide locks where appropriate and that all users actually keep items locked up as requested. The two problems are clearly intertwined. You will never achieve any appreciable level of environmental security unless you address both aspects of the problem.

The security manager is all too aware that a chain will be broken at its weakest link. Typically, environmental weaknesses in security are the ones most readily found and exploited by would-be penetrators. Imagine you had criminal intentions and knew you could easily steal some valuable software by removing a small reel of magnetic tape from your workplace. Would you bother to learn other user's passwords or work to change the file protection on that software, so that you could obtain a copy from an online disk? The path of least resistance would involve the weakness in the physical security. It is fair to say that environmental weaknesses loom as the greatest source of security breaches. However, since the focus of this book is the security features that the VAX/VMS system software provides, you will find very little additional discussion in this guide on how to enhance your environmental security.

Numerous other books available from outside sources discuss environmental security. The security manager at any site must stay current with industry standards and practices in the area of environmental security. Each manager should take an active and determined role in establishing security policy and enforcing it. Without a consistent and thorough approach to the environmental aspects of system security, there is little point in implementing any of the additional suggestions in this guide.

A final environmental concern that this book cannot address is the vulnerability inherent in the use of communications networks. In particular, users with Ethernets should think about what is going over the Ether, who has access to it and what measures are appropriate to protect that information. Beyond that, DIGITAL cannot currently offer products that would make the communications environment truly secure.

---

### 1.3 Levels of Security Requirements

Each site will have unique security requirements. Some sites may be satisfied with minimal measures since they are able to tolerate certain forms of unauthorized access with little adverse effect. At the other extreme are those sites that cannot tolerate even the slightest probing. Among such sites would be strategic military defense centers. Somewhere in between are many commercial sites, such as banks.

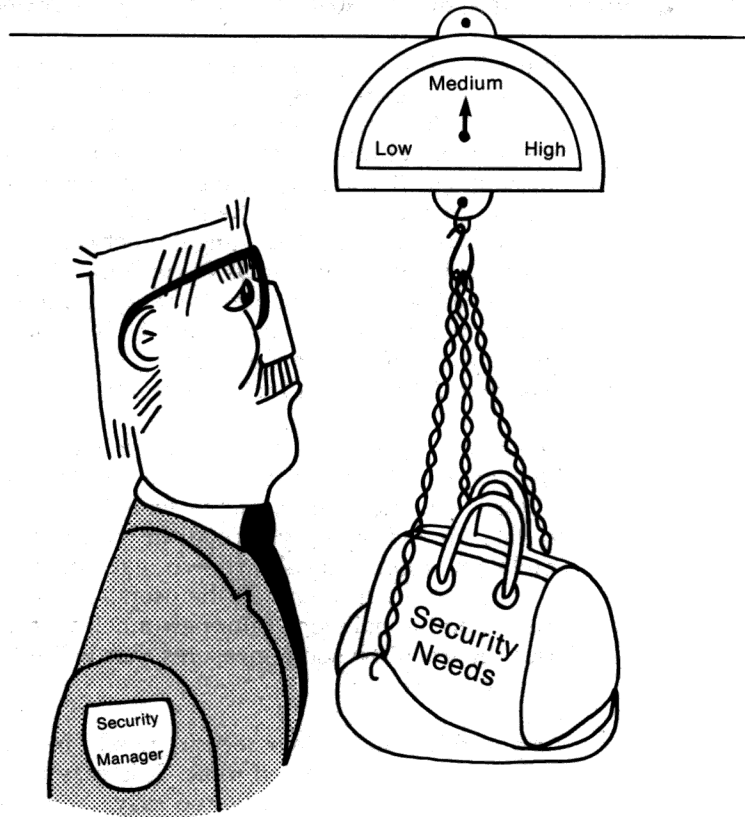
Before each security manager proceeds with the remaining chapters of this book, some difficult questions should be addressed. It may be that you are actually content with your present degree of security and would not take the time or energy to implement additional measures correctly. If so, you should abandon this guide and apply your time elsewhere.

There are some questions you might use to ascertain the extent of your security requirements. Table 1-2 presents some hypothetical events and asks you to respond yes or no according to whether or not you think your site could tolerate the event. As you rank your ability to tolerate the occurrence of any of the events listed in Table 1-2, you will find your answers help categorize your security needs as low, medium, or high.

**Table 1-2 Event Tolerance As A Measure of Security Requirements**

Question: Could You Tolerate The Following Event?	Level of Security Requirements Based on Toleration Responses		
	Low	Medium	High
A user knowing the images being executed on your system	Y	Y	N
A user knowing the names of another user's files	Y	Y	N
A user accessing the file of another user in the group	Y	Y	N
An outsider knowing the name of the system just dialed into	Y	Y	N
A user copying files of other users	Y	N	N
A user reading another user's electronic mail	Y	N	N
A user writing data into another user's file	Y	N	N
A user deleting another user's file	Y	N	N
A user being able to read sections of a disk that might contain various old files	Y	N	N
A user consuming machine time and resources to perform unrelated or unauthorized work, possibly even playing games	Y	N	N

## Introduction



ZK-2058-84

Do not try to use Table 1-2 too literally. Instead, use it to spot trends in your answers. You will find that if you can tolerate most of the events depicted, your security requirements are quite low. If you find your answers about equally mixed between yes and no, your requirements are in the medium to high range. Those sites that are most intolerant to the events, generally have very high levels of security requirements.

Pondering these questions should help you discern the level of security your site requires. Since you probably considered whether or not you could recover if any of the events occurred, be certain you are not projecting a weakness in operations or recovery procedures as a security problem instead. You may find your security needs are not as great as you thought.

---

## **1.4 General Considerations for Security Implementations**

The following chapters pursue each of the security topics according to increasing security impact, which typically increases the work of the users. If your security concerns are in the minimal range, you may not want to implement all the suggestions. If you must strive for maximum security, you will need to pursue nearly all the suggestions. Those at sites with security requirements that range in between these extremes may want to prioritize their areas of concern. Once the priorities are clear, sites can apply time and energy to protecting those areas where unauthorized access would be least acceptable, while ignoring or deferring enhancements that would provide the least amount of return for the investment.

Remember, all security improvements include a cost to the site. The cost may be difficult to measure in dollars and cents, but it will exist. Some security enhancements may slow down access to data, thus delaying your staff in their routine jobs. Other implementations may slow down machine operations, affecting system performance and all the users. Still other measures may require additional steps from personnel, with attendant delays. And there will be some measures that will introduce all these types of costs.

As you think about your security needs, be realistic. Security is an emotionally charged topic. It is easy to justify security measures. In fact, it may be too easy. You may become so overzealous in the pursuit of system security that you adopt too many measures. The most secure systems are predictably the hardest to use and the least friendly to users. They also frequently prove to be the slowest, as a result of delays in manual operations, as well as burdensome machine operations. This guide will try to alert you to both the advantages and disadvantages of each measure. The astute security manager will weigh the pros and cons carefully before attempting implementations. Clearly, not all measures are for every site.

A final word: VAX/VMS provides you with the basic mechanisms to control access to the system and its data by authorized users. It also provides the monitoring tools to ensure that the authorizations are observed. Ultimately, the security of your operation rests in how you apply the security features in the design of your application.

## Introduction

A large number of computer crimes are committed without any violation of the operating system's security controls. Rather, the crimes are committed by authorized users making use of only their authorized access to data. This is particularly true of the commercial world. Thus, while the VAX/VMS security controls serve to protect you from unauthorized persons outside your organization, and can enforce complex patterns of authorizations among your users and data, only you can ensure that your data is not abused by those persons who have been authorized access. You accomplish this in two ways. You build appropriate supervisory controls into your application—controls that are specific to its operation. You also design your application carefully to minimize the opportunity for abuse.





# 2

---

## Overview

---

---

### 2.1 Introduction

This chapter presents the overall concepts that guided the design and implementation of the security features and mechanisms that are incorporated in VAX/VMS. This chapter should help you understand both the structure of VAX/VMS as a secure system and the reasons why individual design decisions were made. Subsequent chapters of this guide present details about the security features and their use. This chapter, on the other hand, should help readers think about security as a whole, so that they can adopt complete and consistent ways of using the features that VAX/VMS offers.

The chapter begins with a description of the *reference monitor concept*, an overall model for system security that governed the security design of VAX/VMS. Next, it outlines the relationship of mechanisms in VAX/VMS to the components of the reference monitor. Finally, the chapter closes with some cautions on the use of VAX/VMS to achieve the security of information.

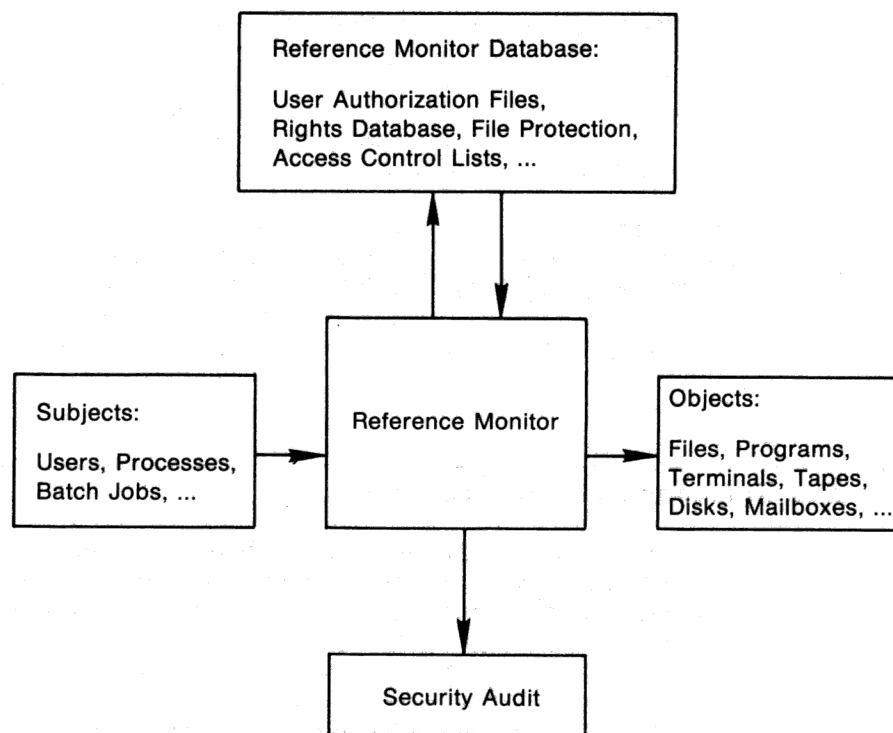
---

### 2.2 The Reference Monitor Concept

In the late 1960s, a great deal of research and development was dedicated to the problem of achieving security in multiuser computer systems. Much of the development work involved attempts to find "all the things that could go wrong" with a system's security, and then to correct those flaws one by one. It rapidly became apparent to the researchers that such a process was a never-ending one, and that effective system security could only result from a basic model of the structure of a secure computer system. The reference monitor concept presented here was proposed as such a model and, after much debate, has gained fairly wide acceptance.

The reference monitor concept depicts a computer system in terms of subjects, objects, an authorization database, an audit trail, and a reference monitor mechanism. Figure 2-1 shows the relationship of these elements. *Subjects* are active entities that gain access to information on behalf of people. *Objects* are passive repositories of information to be protected. The *authorization database* defines the system's security requirements by revealing which subjects (acting on behalf of users) can have which kinds of access to objects (that contain information). The *audit trail* acquires a record of access attempts, whether successful or not, as required by the authorization database.

**Figure 2-1 Reference Monitor Diagram**



ZK-2017-84

## Overview

The reference monitor mechanism enforces the security rules, by mediating the creation of subjects, granting subjects access to objects according to the requirements of the database, and recording events as necessary in the audit trail. In an ideal system, the reference monitor mechanism is required to meet the following three requirements:

- 1 Mediate every attempt by a subject to gain access to an object.
- 2 Provide a tamperproof database and audit trail that are thoroughly protected from unauthorized observation and modification.
- 3 Remain small, simple and well-structured so that its effectiveness in enforcing security requirements can be assured.

A system that fully met all the requirements of the reference monitor model would be very secure indeed. As a matter of fact, these are the requirements proposed for systems that will be secure even against penetration. (In such systems, the reference monitor is implemented by a security-related subset, or *security kernel*, of the operating system). While VAX/VMS is not such a system, its interface to users and system managers does mirror the basic structure dictated by the reference monitor concept. Experience shows that incorporating such a structure is the best way to build a system that is resistant to probing and to many attempts at penetration.

---

### 2.3 The Reference Monitor and VAX/VMS

As the previous discussion suggests, the security structure of VAX/VMS mirrors the reference monitor concept. The subsections below discuss the components of VAX/VMS that play the roles of subjects, objects, authorization database, audit trail, and reference monitor mechanism.

---

### 2.3.1 Subjects

In VAX/VMS, the process performs the role of the subject, the active element that gains access to information. When a user logs in to use VAX/VMS interactively, or when a batch or network job starts, VAX/VMS creates a process with the identity of the user who logged in or submitted the job. That process gains access to information as the agent for the user in the system.

The two critical aspects of security with respect to processes are:

- Process creation
- Access to information by processes

VAX/VMS manages processes' access to information using its authorization data (described below) and internal mechanisms. However, history shows that the area of process creation is rife with opportunities for security problems. For this reason, many of the security features in VAX/VMS concentrate on the area of process (or subject) creation.

When a user attempts to log in to VAX/VMS, the user provides a *username* (a name that will be given to the resulting process) and a *password*. The password serves as an authenticator that should be known only to the user and to VAX/VMS. Because a short or obvious password is likely to fail this requirement, VAX/VMS incorporates numerous mechanisms that can be invoked by the user or required by the security manager to insure against short or obvious passwords. VAX/VMS is also capable of limiting the number of attempts that an intruder can make to guess a user's password. Briefly, then, the association of the user with a subject (or process) is a critical aspect of system security. VAX/VMS provides numerous mechanisms to help insure the integrity of this association.

As you might imagine, the file of users' passwords is part of the reference monitor's database that must be protected from unauthorized observation and modification. VAX/VMS attempts to meet this requirement by storing the password in a file that is normally protected from general access, the *system user authorization file* (SYSUAF.DAT). VAX/VMS also takes the precaution of storing passwords in an encoded form that could not be used even if stolen.

Once it creates a process, VAX/VMS gives that process a *User Identification Code* or *UIC*. The UIC corresponds to the name of the user who created the process (as authenticated by the user's password). In addition, the UIC indicates the user's membership in a group of users that can correspond to the user's department, project, function, or whatever the security manager chooses. VAX/VMS also has the option of attaching additional information to the process, including the circumstances surrounding the creation of the process and the affiliation of the process' owner with various other groups. This additional information plays a part in the application of the authorization database, which is discussed below.

---

### 2.3.2 Objects

In the reference monitor concept, objects are passive repositories of information. In VAX/VMS there are a large number of objects that are subject to protection. The most basic (and usually most important) objects in a VAX/VMS system are files and directories. VAX/VMS protects files and directories from unauthorized access and provides a variety of mechanisms (outlined in the discussion of the authorization database) for their controlled sharing.

Objects other than files and directories can also be used to store sensitive information. These objects include sections, mailboxes, logical names, event flag clusters, and so on. VAX/VMS provides mechanisms to protect these objects as well, but the facilities for controlled sharing are, in some cases, not as rich as those that apply to files.

---

### 2.3.3 The Authorization Database

In the reference monitor concept, each subject's authorization to gain access to an object is indicated in a single authorization database. In VAX/VMS, the database is distributed and stored in association with the objects that must be protected. For example, the authorization data for a file or directory is stored in the file header for that file or directory.

As the discussion of objects suggested, different objects in VAX/VMS can be shared with differing levels of flexibility. Almost all objects are subject to a *UIC-based protection*. This form of protection specifies whether access is allowed or denied

## Overview

to processes running on behalf of the system management, the user who is owner of the object, other members of the UIC group of the owner, and all other users.

In addition to the UIC-based protection, files and directories can be shared under control of *Access Control Lists*, generally referred to as *ACLs*. ACLs list individual users or groups of users who are to be allowed or denied access to the file or directory. ACLs specify sharing on the basis of UIC as well as other groupings or *identifiers* that can be associated with a process. For example, it is possible to specify that a file should never be read by a process connected to a terminal on a dialup line.

---

### 2.3.4 The Audit Trail

VAX/VMS implements the concept of the audit trail in a very straightforward manner. A terminal can be designated as an audit alarm console and all auditable events are displayed on the console. Some events, such as certain login failures and uses of privilege are always auditable. Other events, such as successful or unsuccessful attempts to gain access to sensitive files, can be selected by users or security managers for auditing. For example, the owner of a sensitive file might create an ACL entry requesting that all accesses to that file be audited.

The audit trail mechanism in VAX/VMS allows users and security managers to record a variety of events. As with all audit trails, however, it is no better than the determination of people to examine it and take responsive action. Thus, users and managers must strike a balance between the desire to record everything, and the practical need to select only the number of events that can be examined, allowing some important events to pass unrecorded.

---

### 2.3.5 The Reference Monitor Mechanism

In VAX/VMS, the VAX/VMS executive performs the role of the reference monitor mechanism. In effect, all the system programs that run in kernel and executive mode help to implement the reference monitor, as do certain user-mode images that run with privilege. While the volume of code in this perimeter is large, DIGITAL does attempt to ensure that none of the code can be used to bypass system security in an unauthorized way.

Some of the privileges that VAX/VMS provides can grant a user the power to modify or subvert the reference monitor mechanism. For example, a process with the BYPASS privilege can gain access to any object without reference to the authorization database. Clearly, such critical privileges should only be granted and used with care and forethought.

Similarly, such privileges as SYSPRV and SECURITY are conferred on users whose processes help maintain the reference monitor mechanism and database. Such users might be responsible for adding new users, for example.

---

## 2.4 Using VAX/VMS Securely

The intent of this chapter is to provide readers of this guide with an overall framework for thinking about system security. As individual users and (especially) system managers think about new systems, applications, and environments, they should attempt to keep the reference monitor concept in mind. The following questions are pertinent:

- How are users associated with subjects? What is the reliability of the authentication mechanism?
- What objects contain sensitive information in this system or application? Is access to those objects controlled?
- Does the authorization database reflect policy? Who is authorized to gain access to sensitive objects? Are adequate restrictions in place?
- Is the audit trail recording enough information or too much? Who will look at it?

## Overview

- What programs are functioning as part of the reference monitor mechanism? What users can modify the mechanism and the authorization database? Is this the desired configuration?

The considerations outlined above, and the underlying reference monitor concept, apply equally well in a time-shared VAX/VMS system, a widespread network, or a single application on a VAX/VMS system that grants access to records in a file or database. VAX/VMS attempts to provide general mechanisms to support system security. However, it is the users and system managers who must apply the mechanisms in an appropriate way.



# 3

---

## Security for the User

This chapter provides information about the security features of VAX/VMS that are of interest to all users, from the novice user to the security manager who has overall responsibility for system security. By reading this chapter and becoming familiar with the additional references for the topics provided throughout, the general user should acquire all the basic information needed to use the system securely. Whether or not users apply this knowledge consistently and accurately while observing the site's specific security policies will make a great difference between a secure system and one that is vulnerable.

---

### 3.1 Logging In to the System

As described in the *VAX/VMS DCL Dictionary*, there is a sequence of actions that you follow to gain access to the system. This sequence is known as *logging in*. Login time is the system's first opportunity to check users who request system access to ensure that they are desirable. Generally users must identify themselves with a user name and password to prove they are authorized to use the system.

Login time is also the system's first chance to impose restrictions, if desired, on the use of the system. The greater the security requirements at a site, the more elaborate the login procedure, and the more restrictions the user may face.

### 3.1.1 Types of Logins

VAX/VMS distinguishes logins into seven classes:

- 1 Local
- 2 Dialup
- 3 Remote
- 4 Network
- 5 Batch
- 6 Detached
- 7 Subprocess

These seven classes of logins each employ one of the two methods of logging in: interactive or noninteractive. *Interactive logins* entail a sequence of steps where the system prompts for information and the user provides it. *Noninteractive logins* are performed by the system and require no interaction from the user during the login. Table 3-1 summarizes which types of logins are interactive and which are noninteractive.

**Table 3-1 Classes and Types of Logins**

Login Class	Type
LOCAL	Interactive
DIALUP	Interactive
REMOTE	Interactive
NETWORK	Noninteractive
BATCH	Noninteractive
DETACHED	Dependent on Parent Process
SUBPROCESS	Noninteractive

Note that the term interactive, as used here differs from an interactive mode process that is defined by the DCL lexical function F\$MODE(). In this section, an **interactive login** is one that includes a dialog with the user that provides VAX/VMS with the username and password, as further

described in Section 3.1.2. An **interactive mode** process is one that is declared to be in communication with a person using `SYSS$COMMAND`, and may or may not be created by an interactive login. However, all interactive logins result in an interactive mode process.

---

### 3.1.1.1 Local Logins

*Local logins* are those performed by a user from a terminal that is directly wired to the central processor. Figure 3-1 in Section 3.1.2 illustrates a local login.

Local logins are always interactive.

---

### 3.1.1.2 Dialup Logins

When you log in on a terminal that uses a modem and telephone line to make a connection to the computer system, you are completing the type of login known as a *dialup login*. You may execute a few additional steps initially, depending on the nature of the terminal concentrator that your system uses, if any. (Since the actual procedure is site dependent, it is not described in this guide. Your security manager can provide you with the necessary details.) However, once you receive the Username prompt, the basic elements of the login remain the same as those depicted in Figure 3-1, in Section 3.1.2.

A dialup login is always an interactive login.

---

### 3.1.1.3 Remote Logins

When you log in to a node over the network, you request that node using the DCL command `SET HOST`. Such a login is known as a *remote login*. The node you reach then immediately asks you for a user name and password, much like the local login illustrated in Figure 3-1, in Section 3.1.2. The various informational messages remain the same, and thus optional.

A remote login is always an interactive login.

---

#### 3.1.1.4 Network Logins

*Network logins* are performed for you when you access files stored in a directory on another node or when you initiate some other type of network task on a remote node. Both your current system and the remote system must be nodes in the same network. You use one of the DCL commands that apply over networks. In the file specification you specify the desired node and an optional access control string, where the access control string includes your username and password for the remote node.

*Proxy logins* represent a special type of network login. With a proxy login you also gain network access, but you do not have to include an access control string to provide the username and password for your network login request. Thus, proxy logins have several security implications. First, passwords are never echoed on the terminal where the request originates, which is very desirable. Second, passwords are not passed between systems where they might be intercepted in unencrypted form. Third, this technique removes the temptation to store passwords in command files that would perform the remote access steps.

Proxy logins are described in detail in Section 3.2.2.

All network logins are noninteractive.

---

#### 3.1.1.5 Batch Logins

A *batch login* is accomplished on your behalf when a batch process that you initiate actually runs. For example, you might submit a job to run after 7:00 p.m. with the following DCL command:

```
$ SUBMIT/AFTER=19:00 PAYROLL.COM
```

When the system prepares to execute PAYROLL.COM, sometime after 7:00 p.m. the day the job is submitted, the batch job controller first logs in to the user's account to gain access to the program. The login is classified as a batch login, and is noninteractive.

The batch job controller does not need to provide a password to effect the login.

---

**3.1.1.6 Detached Process Login**

A *detached process* login can occur as the result of the execution of either the process form of the DCL command RUN or the system service \$CREPRC, where the image specified is SYS\$SYSTEM:LOGINOUT.EXE and the options on the service call or the RUN command specify a detached process.

The creator of a detached process login can specify that its type of login is either interactive or noninteractive.

---

**3.1.1.7 Subprocess Login**

A *subprocess login* can occur as the result of the execution of either the process form of the DCL command RUN or the system service \$CREPRC, where the image specified is SYS\$SYSTEM:LOGINOUT.EXE and the options on the service call or the RUN command specify a subprocess. In addition, a subprocess login results when the DCL command SPAWN executes.

A subprocess login is always a noninteractive login, and it is always created to run under the account of the creator.

---

**3.1.2 Interactive Login Informational Messages**

It may be helpful to review the elements of an interactive login and to discuss which parts of the display the system manager may choose to suppress and why. Figure 3-1 represents a local login from a terminal that is directly connected to the system. For instructional purposes, Figure 3-1 includes examples of most of the informational messages that might appear.

**Figure 3-1 Example of Local Login, Illustrating Messages**

WILLOW - a member of the Arboretum VAXCLUSTER	①	announcement msg
Username: RWOODS		
Password:		
You have the following disconnected process:	②	disconnected job messages
Terminal Process name Image name		
VTA52: RWOODS (none)		
Connect to above listed process [YES]: NO		
Welcome to VAX/VMS Version 4.0 on node WILLOW	③	welcome msg
Last interactive login on Wednesday, 1-AUG-1985 10:20	④	last login msg 1
Last non-interactive login on Monday, 30-JUL-1985 17:39	⑤	last login msg 2
2 failures since last successful login	⑥	last login msg 3
You have 1 new mail message.	⑦	new mail msg
\$		

In the example in Figure 3-1, most of the informational messages are present. You can observe the announcement message at callout ①, the disconnected job messages at callout ②, the welcome message at callout ③, as well as the optional group of last login messages at callouts ④, ⑤, and ⑥. If new mail has been delivered, users may receive a new mail message similar to the one at callout ⑦.

### 3.1.2.1

#### Announcement Message

The *announcement message* typically identifies the node (and, if relevant, the cluster) that you have succeeded in accessing. The announcement message is your first visual indication that you have initiated the login process. The announcement message immediately precedes the Username prompt. Both the appearance of this message and the message content can be controlled by the system and/or security manager.

---

### 3.1.2.2 Disconnected Job Messages

When you succeed in logging in, you may occasionally see messages similar to those in Figure 3-1 at callout ② that report *disconnected jobs*. Such messages indicate that a previous login was interrupted prematurely but is available for reconnection, at your option. These messages can only appear when two conditions exist. First, the terminal where the interruption occurred had been set up (as described below) to prohibit a process from being disconnected when the line sensed a hangup. Second, during a recent prior session your connection to the CPU through that terminal was broken. (The initial setup requires the system manager to perform several steps. The system manager uses the DCL command SET TERMINAL/PERMANENT/DISCONNECT to identify the intended terminals and also specifies parameters to the System Generation Utility to enable *virtual terminals* and establish how long the virtual terminal connection remains.) Generally, you only have a certain limited time, such as the VAX/VMS default value of 15 minutes, to login again and make the reconnection.

The *disconnected job messages* inform you that your process was disconnected at some time after your last successful login. You are given the option of reconnecting to the old process, unless you choose otherwise. Reconnecting can be very helpful if you were in the middle of some work when the connection was lost. If you take the default action or respond to the question with a "yes" answer, you are logged out of your current process just as an automatic execution of the DCL command CONNECT /CONTINUE is performed for you. If you specify "no" in response to the reconnection question, or you delay too long in responding so that a response period timeout occurs, you remain logged into your new process and you lose the ability to connect to the old process.

---

### 3.1.2.3 Welcome Message

Next, once you have logged in, you may find a *welcome message* that indicates the software version of VAX/VMS that is running, and sometimes also includes the node, again at the discretion of management. If management chooses, an entirely different message content may appear. Also, management can choose to suppress the message entirely.

#### 3.1.2.4 Last Login Messages

Immediately following the welcome message, if there is one, you may see any of three messages that provide information about the last successful login. These *last login messages* are also entirely optional. However, they are enabled or disabled as a group. That is, if one message can be displayed, all that are pertinent will be displayed. If disabled, none of the messages will appear.

If the messages are enabled, you will receive from one to three of the following kinds of last login messages:

- Last successful interactive login message—provides the time of the last login that completed for a local, dialup, or remote login. (Note that logins from a subprocess whose parent was one of these types are not included in the count.) An example appears at callout ④ in Figure 3-1.
- Last successful noninteractive login message—provides the time the last noninteractive login successfully completed. Noninteractive logins refer to those of batch or network processes. An example appears at callout ⑤ in Figure 3-1.
- Number of login failures—if any attempts have been made to login with this username and have failed because of an incorrect password, they are recorded in a count that is displayed in this message at the next successful login. Right after the message appears, the bell rings to prompt your attention. (The specification of an incorrect password is the only source of login failure that is counted.) An example appears at callout ⑥ in Figure 3-1.

All three types of login message can be displayed at the next successful login, whereupon the values for the the last successful login and the number of login failures are reset. If your account is never accessed other than interactively, and you never specify an incorrect password in your login attempts, you might never see the second or third type of last login message. However, once your account is accessed noninteractively, that message will reappear unchanged at each successful login until the next noninteractive access causes a change.



---

### **3.1.2.5 Message Suppression**

Sometimes a security manager prefers to suppress the announcement message and the welcome message so that the details of the node or operating system are not immediately revealed. This makes it a little more difficult for an outsider to know the proper login procedure, since login procedures typically vary from operating system to operating system. This is an action usually taken only at the sites that have medium- or high-level security concerns.

The same site that suppresses the welcome and announcement messages for security reasons, is likely to exercise the option of displaying the last login success and failure messages. These messages are valuable to authorized users when checked regularly for unexplained logins and unsuccessful login attempts. At the same time, when an illegal user encounters such messages for the first time, the effect can be disquieting; they show that the system monitors logins.

The disconnected job messages appear less frequently and only under special circumstances. That is, virtual terminals must be enabled on your system and the terminal whose connection was broken prior to a logout during your last session must also have been set up not to disconnect. The security manager can disable this function by changing the setup on terminals and disabling virtual terminals on the system. The ability to reconnect is generally desirable and offers no special problems for system security.

---

### **3.1.3 Introduction to Passwords**

Passwords are commonly requested by systems at login time. VAX/VMS is no exception. Passwords are typically strings of characters that users must specify when they log in to authenticate their identity as the person authorized to use the account. To preserve the secrecy of the passwords, terminals do not echo the password characters as they are entered from the input device. Proper administration of passwords is critical to the security of a system.



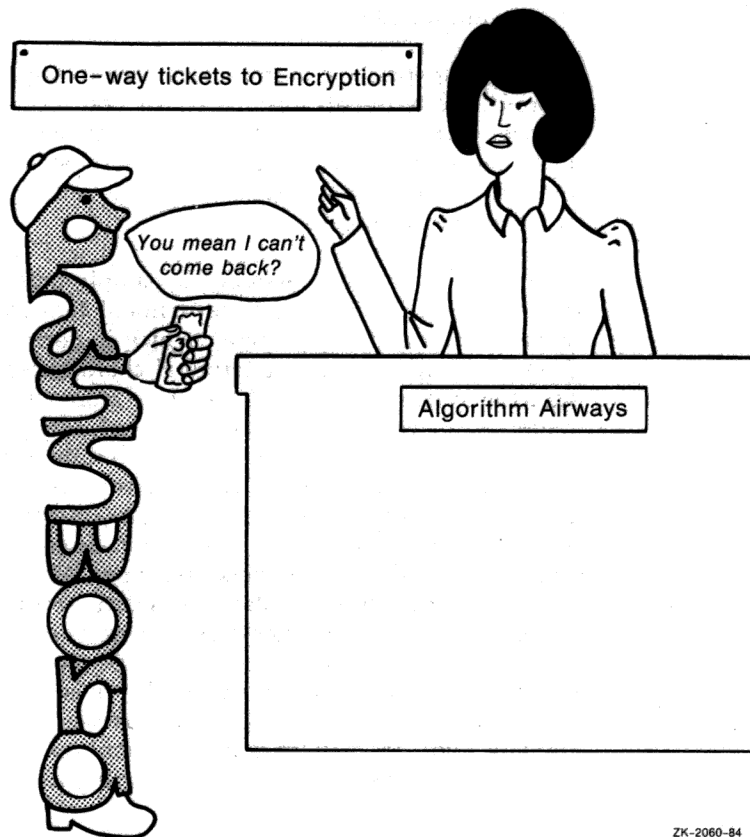
ZK-2059-84

There are various types of passwords that you may encounter on the VAX/VMS system. Most users will need to provide a *user password* when they login. Other users will also need to provide a *system password* to first gain access to a particular terminal before logging in with their user's password. And still other users on systems with high security requirements will find themselves concerned with *primary passwords* and *secondary passwords*. Each of these types of passwords are described in the following sections.

Users are naturally curious about how the system stores passwords and validates them. The typical intent is for each password to remain secret and authorized uniquely for its user. If the system keeps passwords in a system table, there is the possibility an outsider might gain access to the table and learn

all the passwords. To circumvent this possibility, VAX/VMS applies a *one-way encryption algorithm* to all passwords as it stores them.

Encryption refers to a method of encoding information in an effort to conceal it from undesirable sources. One-way encryption algorithms are effective in preventing the result from being decrypted, since their distinctive characteristic is that they do not use a key. Thus, even if a malicious user succeeds in obtaining the encryption algorithm and the encoded password, that user could not deduce the actual password that was input, except by trying all possible input values.



ZK-2060-84

In practice, whenever a user specifies a password, VAX/VMS always runs the submitted password through the encryption algorithm before attempting to match the derived value with the stored value. If the two encoded values match, VAX/VMS grants access to the user.

---

### 3.1.3.1 System Passwords

As indicated earlier, system passwords control access to particular terminals. They are optionally required by the security manager. Sometimes they are desirable simply as another level of security, but most often they are selected as a means of controlling access to terminals that might be targets for unauthorized use.

If you must specify a system password to log in to one or more of the terminals designated for your use, your security manager will tell you. Your security manager will also provide information on the current value of the system password, how often it changes, and how to obtain the new system password when it does change.

You specify a system password by simply pressing the carriage return key (RETURN) until the terminal *autobauds* and responds with the recognition character, which is normally a bell. (Autobauding is a settable terminal characteristic that requests that the system detect, and therefore automatically set, the baud rate of your terminal from a few RETURN keys that you initially send to the system.) If your terminal has been set with the /NOAUTOBAUD characteristic, you will only press the RETURN key once. At that point you type the system password followed by a RETURN key. There will be no prompt and no echo of the characters you type. When you succeed in entering the correct system password, you will receive the system announcement message, if there is one, followed by the username prompt. If you fail to specify the correct system password, you can try repeatedly. There is no notification given that you have entered an incorrect password. Thus, you might initially think the system is malfunctioning, unless you know that a system password is required at that terminal.

### 3.1.3.2

#### User Passwords

If you view accounts on a system from the standpoint of whether or not they require user passwords and whether or not users can change their passwords, you find there are four types of accounts available on VAX/VMS systems:

- *Open accounts* that require no password; the password is said to be null. Users of open accounts are the only users who will not encounter user passwords in one form or another. With an open account, the user is not prompted for a password and can begin entering commands immediately.
- *Captive accounts* that permit very limited operations and may require a password but typically tightly control what commands the user can implement. Since the DCL command SET PASSWORD is unlikely to be in the command repertoire, the captive user rarely changes the password.
- Accounts that require passwords but prohibit the user from changing the password. The password is said to be locked (accomplished through the LOCKPWD flag in the User Authorization File). One purpose of locking the password is to allow the security manager to supply (and periodically change) a password, to ensure that the user does not change it, possibly to a less secure password.
- Accounts secured with passwords that the user or security manager change from time to time. These are the most common kinds of accounts, and will be treated as the norm throughout this guide.

It is important to know which type of account you have so that you can use passwords appropriately.

Typically, when you learn an account has been created on the system for you, you are told whether or not a user password is required. If user passwords are in effect, you will likely be told to use a specific password for your first login. This password has been placed in your record in the User Authorization File (UAF) along with other pertinent information about how your account can be used.

## Security for the User

A common practice is to use your first name as your first password. Unfortunately, this practice is so well known that is very undesirable from a security standpoint. Although you will be requested to change your password as one of your first actions after logging in, under the scenario of a first-name password your account remains highly vulnerable until you do so. If there is a time lapse from the time your account is created until your first log in, other users might log in to your account successfully before you, thus gaining a chance to damage the system. Similarly, if you neglect to change the password, or are unable to do so because of inexperience or some other problem, the system remains vulnerable. Just how much damage can be done depends largely on what other security measures are in effect.

When security matters, it is never good practice to have accounts on the system where the password can be easily guessed. Common password choices that you should specifically **avoid** include:

- Your name
- The name of a family member or loved one
- The name of a pet
- The make of your favorite type of automobile
- The name of your home town
- The name or make of your boat
- Any name associated with your work, such as your company, special project, group, and so forth
- Any other item that bears a strong personal association to you

At the time your account is opened, you should also be told a minimum length for your passwords and whether or not you will be able to choose your new password or whether you must let the system automatically generate the password for you.

### **3.1.3.3 Changing Your User Password**

You change your password with the DCL command SET PASSWORD. This command supports two modes: manual changes provided by the user and automatic generation of new password choices by the system. Sometimes it is your choice as to whether or not you want to use the automatic password generation feature, and sometimes your system manager will require you to use this feature. Automatic password generation is described in more detail below.

#### **User-Selected Passwords**

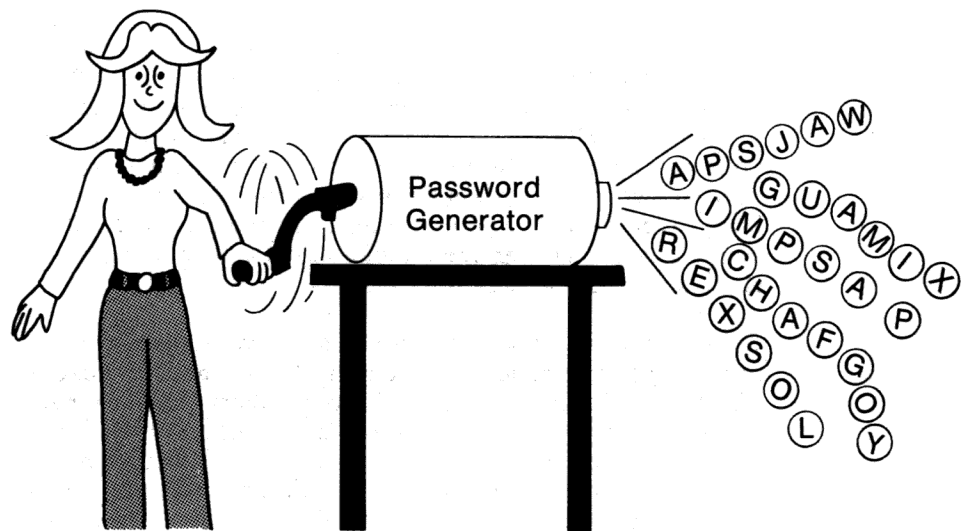
Assume for the moment that you are interested in changing your password to a new password of your own choice. You will have learned the minimum number of characters required for your VAX/VMS passwords. This characteristic is part of your user authorization record, so that when you designate a new password, the system enforces your minimum password length requirement. You can enter a password choice that is equal to or longer than the minimum, but any shorter password choices will be rejected.

Once you have a new password in mind that satisfies the minimum password length requirement, you simply invoke the SET PASSWORD command with no qualifiers. The command will prompt you to provide the old password and then will request that you enter the new password. As a final step, the system asks you to enter the new password one more time, for verification. If you fail to enter the old password correctly or do not enter the same password twice as the new password, the password is not changed. If you succeed in these three steps, there is no notification, but rather the command terminates, and your password is changed. (Section 3.1.3.5 describes in more detail the consequences of entering a new password choice that does not meet the minimum password length requirement.)

#### **Automatically Generated Passwords**

If your system's security manager has decided that you must let the system generate the password for you automatically, you will find that you must use the DCL command SET PASSWORD/GENERATE to change your password. (If you attempt to use the SET PASSWORD command without the /GENERATE qualifier, you will find it is automatically provided.)

*Automatic password generation* produces a list of password choices for the user where each of the passwords consists of random sequences of characters that resemble English words to make them easy to remember, but unusual enough to be difficult for outsiders to guess. Furthermore, the conniving and determined user who wrote a program to test every word in the English dictionary as a possible password choice, would be very likely to fail.



ZK-2061-84

You will find, too, that while the length of the passwords generated for you satisfies the minimum, the length is not necessarily always the same. (This adds another element of difficulty for the probing outsider who is trying to guess passwords.)



## Security for the User

The following example illustrates a user requesting automatic password generation. The minimum password length for this user has been set to 6 in the UAF.

```
$ SET PASSWORD/GENERATE=8
```

```
Old password:
```

apsjawpha	aps-jaw-pha
oorsoult	oor-soult
guamixexab	gu-a-mix-ex-ab
impsapoc	imps-a-poc
ukchafgoy	uk-chaf-goy

```
Choose a password from this list or press RETURN to get a new list
```

```
New password:
```

```
Verification:
```

```
$
```

In the example above, the user requests the automatic password generator to provide password choices with a minimum length of eight. (This length more than satisfies the minimum of six characters that the user's UAF record dictates.) The user correctly specifies the old password and then enters a RETURN key. The system responds with a list of five password choices, ranging in length from eight to ten characters each.

Notice in the example that to the right of each password choice is a representation of the same word divided into its syllables. (To choose a password, attempt silent pronunciations as suggested by the syllables. Usually the password that is easiest to pronounce is easiest to remember, and therefore, the best choice.)

Next, the system reminds the user that it is possible to request a new list by pressing the RETURN key in response to the prompt for a new password that follows. However, in this case, the user enters one of the first five possible passwords, followed by the RETURN key. The system recognizes that this password is one provided by the automatic password generator and responds with the Verification prompt. The user enters the new password again followed by a RETURN key. The system changes the password and responds with the system or DCL prompt.

On systems where you are not required to use the password generator, you are strongly encouraged to use it on your own. This is a very good practice that promotes the security of your system, and it has very few drawbacks. Two minor disadvantages found with automatic password generation include the introduction of a small amount of additional

processing time for the software to generate the password and the possibility that users may not remember their password choices. However, if you dislike all the password choices in your list or think none will be easy to remember, you can simply request another list.

A more serious potential drawback is the possible disclosure of the password choices from the display the command produces. If an insider comes upon this display, it is guaranteed that in five tries or less that user can gain access to your account! To protect your account, perform automatic password changes as unobtrusively as possible, preferably in private. Furthermore, always think about the remaining evidence of the change. If you perform the change on a video terminal, erase the display of the password choices from the screen after the command completes. If you use a printing terminal, properly dispose of all hardcopy output. If you realize afterwards that you failed to protect your password in these ways, promptly change your password. Depending on site policy or your own judgement concerning the length of time your account was exposed, you might decide to notify your security manager that a security breach could have occurred through your account. The most important step is to change the password, but sometimes it is also helpful to alert the security manager to watch for evidence of possible tampering.

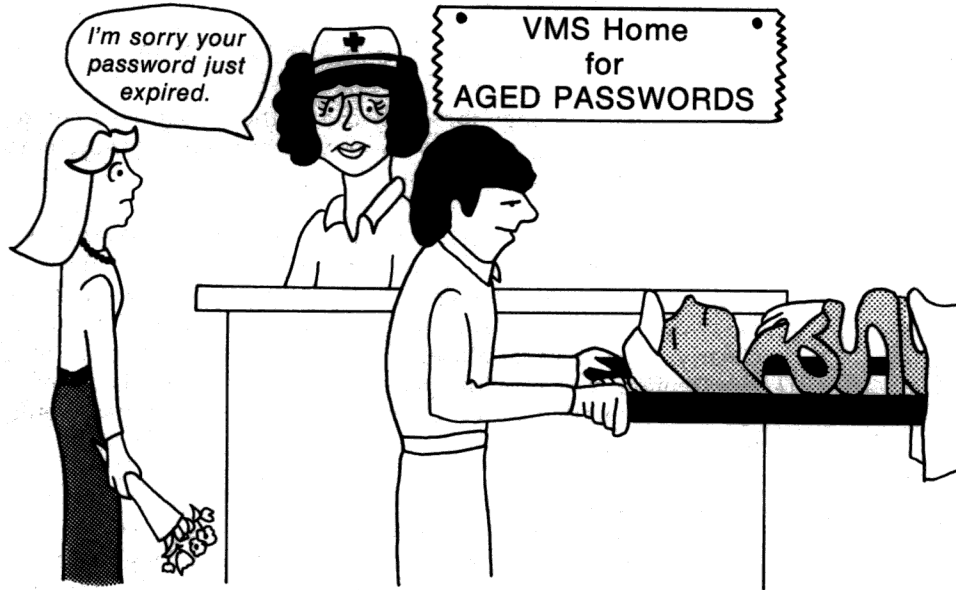
**Note:** The password generator uses basic syllabic rules to generate words, but has no real knowledge of any language. As a result, it could produce words that are offensive. Efforts were made to filter out a number of obviously offensive words. However, if an objectionable word appears, please accept an apology from DIGITAL and select another password.

There is no restriction on how many times you can change your password in a given period of time. On the other hand, there is a maximum period of time that you can retain the same password. This maximum period is dictated by the password lifetime characteristic that is set by management in your UAF record.

**3.1.3.4**

**Password Expiration Time**

When a password reaches the end of its password lifetime, it is said to have expired. This security feature helps to remind and force users to change their passwords with regular frequency. Changing passwords on a regular basis promotes system security.



ZK-2062-84

As you approach the expiration time of your password, you receive an advance warning message. The message first begins to appear each time you login within the remaining five days prior to the expiration. The message appears immediately below the new mail message, if any, and it sounds the bell character on your terminal to attract your attention. The message indicates that your password is expiring, as in the sample message below:

WARNING---Your password expires on Thursday 19-JUL-1985

You must change this password within the next five days or you will find yourself with one last chance to log in and change the password. Thus, should you fail to receive five days advance notice (possibly as a result of a vacation or unplanned absence), or you let the five days pass, you at least have one last chance to make the change. You will receive the following type of message, followed by the sound of the bell character:

WARNING---Your password has expired; update immediately with SET PASSWORD!

If you forget to change the password or the system fails before you have the opportunity to do so, you must see your system manager to regain access.

Note that you will be unable to specify your old password as your new password. This requirement fosters true changes in passwords, by discouraging users from performing a token password change that sets the password to its same old value. Clearly, the determined user can circumvent the intent of this requirement with multiple password changes, but it is hoped that if users understand the importance of changing passwords, they will comply.

### 3.1.3.5

#### Minimum Password Lengths

Your security manager can establish a minimum length for your passwords by specifying a value in your UAF record. When you consider that increasing the length of the password increases the number of possible combinations of letters and digits, you begin to see the security motivation for this requirement. Longer passwords increase the difficulty met by an outsider who may try to guess passwords or use a program to derive all the combinations. As a result, you will find that the VAX/VMS system encourages minimum password lengths in the range of 6 through 10 characters. If you can select your own password, you can opt for a password as long as 31 characters, but generally you will find that entering such long passwords becomes too cumbersome and error-prone to be practical.

Automatic password generation always provides password choices that are acceptably long. However, if you are able to select your own passwords, be sure to observe the minimum

## **Guide to VAX/VMS System Security**

Order Number: AA-Y510B-TE

**July 1985**

This guide describes the security features available through the VAX/VMS operating system. It explains the purpose and proper application of each feature in the context of specific security needs.

This manual contains updated information. Technical changes are indicated in the left-hand margin by change bars (■) for additions, and bullets (●) for deletions.

### **Revision/Update Information:**

This document supersedes the *Guide to VAX/VMS System Security* Version 4.0

### **Software Version:**

VAX/VMS Version 4.2

**digital equipment corporation  
maynard, massachusetts**

---

**July 1985**

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital Equipment Corporation or its affiliated companies.

---

Copyright ©1985 by Digital Equipment Corporation  
All Rights Reserved

The postpaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DEC	DIBOL	UNIBUS
DEC/CMS	EduSystem	VAX
DEC/MMS	IAS	VAXcluster
DECnet	MASSBUS	VMS
DECsystem-10	PDP	VT
DECSYSTEM-20	PDT	
DECUS	RSTS	
DECwriter	RSX	<b>digital</b>

ZK-2835

---

**HOW TO ORDER ADDITIONAL DOCUMENTATION  
DIRECT MAIL ORDERS**

**USA & PUERTO RICO\***

Digital Equipment  
Corporation  
P.O. Box CS2008  
Nashua, New Hampshire  
03061

**CANADA**

Digital Equipment  
of Canada Ltd.  
100 Herzberg Road  
Kanata, Ontario K2K 2A6  
Attn: Direct Order Desk

**INTERNATIONAL**

Digital Equipment  
Corporation  
PSG Business Manager  
c/o Digital's local  
subsidiary or  
approved distributor

In Continental USA and Puerto Rico call 800-258-1710.

In New Hampshire, Alaska, and Hawaii call 603-884-6660.

In Canada call 800-267-6215 (in Ottawa-Hull 613-234-7726).

\*Any prepaid order from Puerto Rico must be placed with the local Digital subsidiary (809-754-7575).

Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Northboro, Massachusetts 01532.

---

This document was prepared using an in-house documentation production system. All page composition and make-up was performed by T<sub>E</sub>X, the typesetting system developed by Donald E. Knuth at Stanford University. T<sub>E</sub>X is a registered trademark of the American Mathematical Society.

---

# Contents

---

PREFACE	xv
---------	----

---

SUMMARY OF TECHNICAL CHANGES	xxiii
------------------------------	-------

---

---

CHAPTER 1 INTRODUCTION	1-1
------------------------	-----

---

1.1	TYPES OF COMPUTER SECURITY PROBLEMS	1-2
1.1.1	User Irresponsibility	1-2
1.1.2	User Probing	1-3
1.1.3	User Penetration	1-4
1.1.4	Characterizing the Perpetrators	1-4
1.2	THE SECURE SYSTEM ENVIRONMENT	1-5
1.3	LEVELS OF SECURITY REQUIREMENTS	1-7
1.4	GENERAL CONSIDERATIONS FOR SECURITY IMPLEMENTATIONS	1-10

---

---

CHAPTER 2 OVERVIEW	2-1
--------------------	-----

---

2.1	INTRODUCTION	2-1
2.2	THE REFERENCE MONITOR CONCEPT	2-1
2.3	THE REFERENCE MONITOR AND VAX/VMS	2-3
2.3.1	Subjects	2-4
2.3.2	Objects	2-5
2.3.3	The Authorization Database	2-5
2.3.4	The Audit Trail	2-6
2.3.5	The Reference Monitor Mechanism	2-7

---

2.4	USING VAX/VMS SECURELY	2-7
-----	------------------------	-----

---

<b>CHAPTER 3</b>	<b>SECURITY FOR THE USER</b>	<b>3-1</b>
------------------	------------------------------	------------

---

<b>3.1</b>	<b>LOGGING IN TO THE SYSTEM</b>	<b>3-1</b>
<b>3.1.1</b>	<b>Types of Logins</b>	<b>3-2</b>
3.1.1.1	Local Logins • 3-3	
3.1.1.2	Dialup Logins • 3-3	
3.1.1.3	Remote Logins • 3-3	
3.1.1.4	Network Logins • 3-4	
3.1.1.5	Batch Logins • 3-4	
3.1.1.6	Detached Process Login • 3-5	
3.1.1.7	Subprocess Login • 3-5	
<b>3.1.2</b>	<b>Interactive Login Informational Messages</b>	<b>3-5</b>
3.1.2.1	Announcement Message • 3-6	
3.1.2.2	Disconnected Job Messages • 3-7	
3.1.2.3	Welcome Message • 3-7	
3.1.2.4	Last Login Messages • 3-8	
3.1.2.5	Message Suppression • 3-9	
<b>3.1.3</b>	<b>Introduction to Passwords</b>	<b>3-9</b>
3.1.3.1	System Passwords • 3-12	
3.1.3.2	User Passwords • 3-13	
3.1.3.3	Changing Your User Password • 3-15	
3.1.3.4	Password Expiration Time • 3-19	
3.1.3.5	Minimum Password Lengths • 3-20	
3.1.3.6	Selecting Secure Passwords • 3-21	
3.1.3.7	Primary and Secondary Passwords • 3-21	
3.1.3.8	Avoiding Programs that Steal Passwords • 3-23	
3.1.3.9	Protecting Your Password • 3-25	
3.1.3.10	Summary of Password Guidelines • 3-27	
<b>3.1.4</b>	<b>Account Expiration Times</b>	<b>3-28</b>
<b>3.1.5</b>	<b>Causes of Login Failures</b>	<b>3-29</b>
3.1.5.1	System Password Failures • 3-30	
3.1.5.2	Login Class Restrictions • 3-31	
3.1.5.3	Shift Restrictions • 3-31	
3.1.5.4	Dialup Login Failures • 3-32	
3.1.5.5	Breakin Evasion Has Been Activated • 3-32	
3.1.5.6	Summary of Login Failure Sources • 3-33	

---

<b>3.2</b>	<b>NETWORK SECURITY CONSIDERATIONS FOR USERS</b>	<b>3-34</b>
<b>3.2.1</b>	<b>Network Access Control Strings</b>	<b>3-34</b>
<b>3.2.2</b>	<b>Proxy Logins</b>	<b>3-35</b>



## Security for the User

password length. Otherwise, your choice will be rejected by the SET PASSWORD command with the following error message:

```
%SET-E-INVPWDLEN, invalid password length - password not changed
```

Minimum password lengths can promote password choices that are not easily guessed or derived. They offer more advantages than disadvantages on most systems.

### 3.1.3.6

#### Selecting Secure Passwords

Previous sections have described the importance of length and the avoidance of English words in selecting your passwords. (Actually, you should avoid words in the national language of your country. English is only specified here, because this guide originates in the English language.) Your goal should be to avoid words that are readily found in a dictionary. In that way, your password choice is less subject to discovery by a program that successively enters the words in the dictionary, searching for the one that produces a successful login.

It turns out that in the quest for secure passwords, the content you choose for your password is more important than the length. Using digits as well as letters provides the most secure passwords. For example, if you could choose a six-character password using letters only, you would find there are 300 million combinations. However, when you allow that same six-character password to include digits, you increase the number of combinations to 2 billion. When you consider that the total number of English words **in common use**, of all lengths, is less than 2000, you see more clearly why you should avoid words in the dictionary. If you are allowed to choose your own passwords, consider including digits in your selections.

### 3.1.3.7

#### Primary and Secondary Passwords

VAX/VMS can provide an additional level of security on user accounts by requiring the use of secondary passwords in addition to primary passwords. Your security manager decides whether or not to adopt this practice for your account at the time the account is created. When primary and secondary passwords are required, it is impossible to log in without first specifying both passwords correctly in succession.

## Security for the User

In some cases one user knows both passwords and uses them to log in. However, the more typical case involves two users, where the primary user does not know the secondary password. This arrangement is designed to facilitate controlled logins. By requiring the presence of a supervisor or other additional key person with you at the terminal at login time, there is added assurance that your true identity is known. This additional step should prevent anyone else from using your password to gain access to your account.

Another purpose for dual passwords is controlling the actions taken after a login. Users cannot act alone until the holder of the secondary password leaves. For certain applications, it may be desirable for another person to remain present the entire time the account is in use, until a logout is completed. Keeping the second person in attendance cannot be enforced by the software, but the software can ensure that the login is not accomplished until the key person is at least present.

A login requiring primary and secondary passwords might appear as follows:

```
WILLOW - a member of the Arboretum VAXCLUSTER
Username:
Password:
Password:
Welcome to VAX/VMS Version 4.0 on node WILLOW
$ RWOODS
```

The only difference between this login and one that requires only a primary password is the appearance of the second password prompt. When the second password prompt appears, the user who knows the secondary password must enter it. As with a single password login, there is a limited amount of time allotted for the entire login. If the entry of the secondary password is not completed in time, the login will time out.

This is clearly a time consuming and inconvenient practice that is only justified at sites with maximum security requirements or for supersensitive accounts. An example of a supersensitive account that justifies dual passwords might be one that bypasses normal access controls to permit emergency repair to a database.

There are some additional considerations when it comes time to change the primary and secondary passwords. Each password can be changed independently, at different times, but both are subject to the same change frequency, since they share the same password lifetime. Also, the minimum password length applies to both passwords.

The procedure for changing the primary password is basically the same as that described in Section 3.1.3.3. The only difference is that you use the SET PASSWORD/SECONDARY command to change the secondary password. VAX/VMS prompts you to specify the old (secondary) password and the new (secondary) password, just as in the procedure for changing the primary password.

### 3.1.3.8

#### Avoiding Programs that Steal Passwords

Beware of approaching an already turned on terminal and attempting a login. You just might be revealing your password to a program that is specially designed to steal passwords! This precaution is particularly relevant when you log in in a public terminal room.

In this scenario, a clever programmer has implemented a special program that displays an empty video screen, a screen that appears to show the system has just been initialized after a crash, or a screen that shows a phony logout. The program waits for an unsuspecting user to attempt a login. The program will run through the normal login sequence with you so that you think you are entering your username and password to the system. However, once the program receives this key information and passes it on to the perpetrator, it will display a login failure. You may think you mistyped your password and be none the wiser that you have just revealed this vital information. Programs that steal passwords in this fashion are commonly referred to as *password grabbers*.

You should try to eliminate this possibility. Your security manager will instruct you on how to proceed. Frequently you will be advised to press the BREAK key before pressing the RETURN key. Pressing the BREAK key invokes the *secure terminal server* feature for the terminal, if it has been enabled by the security manager. The purpose of the secure server is to ensure that the VAX/VMS login program is the only program able to receive your login.



ZK-2063-84

Never walk away from a terminal after you have logged in. You may return and be misled into thinking the system failed and then came back up again, only to have had the devious insider load the password stealing program into your area to wait eagerly for you to fall into its trap. Even a terminal that displays an apparently valid LOGOUT message may not reflect a process in the logged out state that you would normally expect.

Finally, check your last login messages routinely. Remember, the password stealing program cannot increase the login failure count, even though it may portray its exit as a login failure to you. So, be particularly alert for login failure counts that either do not appear following your failure, or that are one less than the number you experienced. If you observe this, or any

other anomalous failure during a login, change your password immediately and notify your security manager.

### 3.1.3.9

#### Protecting Your Password

Illegal system accesses involving the use of a correct password are more often traced to disclosure of the password by its owner than to surreptitious discovery. Thus, your responsibilities regarding your password include not only changing it frequently, but keeping it secret. You should never post it in your office where it might be observed or discovered. You should not give it out to friends, store it in a file, or send it in a mail message.

One of the tricks used by browsers who succeed in accessing a system is to search inadequately protected files for character strings that are likely to appear in conjunction with actual passwords. One of the possibilities is the three-character string that is specific to network access control strings—a quotation mark followed by two colons ("::). This string immediately follows the username and password specification for network file accesses. Another possibility worth searching for is just the noun password. For example, a careless user generally reveals the actual password near the string "password" in a sentence much like this one:

My password is GOBBLEDYGOOK.

The browser hopes that even if not much can be done from the first account that is accessed, some other passwords can be gleaned to move on to bigger and better things at the next account, and so on.



ZK-2064-84

Persons on password hunts may try another trick to take advantage of a user's natural tendency to select one password for use on multiple systems. If the password hunter succeeds in learning a password for one system, the next step is to try that password out on every other system where the same user has an account. Again, the account that first reveals the password may not hold very much of interest, but there is always the possibility that another one of the user's accounts will yield either more information or more privileges, ultimately leading to far greater things.

If you hold accounts on multiple systems, there are several special considerations for your password as outlined below:

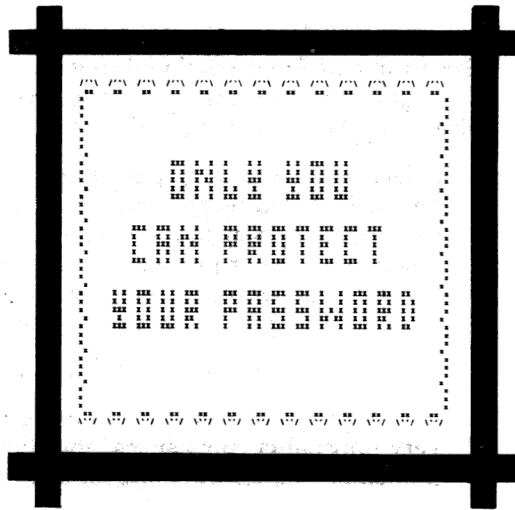
- If any of the systems requires high security, use a unique password on that system.
- If any of the systems uses non-DIGITAL computers, use unique passwords on your accounts.
- If the systems are either all VAX/VMS systems or employ equivalently good password encryption, you can use the same password for each system, provided you observe all the other guidelines regarding good password choices.
- If any system does not use password encryption, select a unique password for that system to prevent the likelihood that a browser who discovers your unencrypted password can gain access to your other accounts.

### 3.1.3.10 Summary of Password Guidelines

To summarize, you can best protect your password by observing the following guidelines:

- Select reasonably long passwords that cannot be easily guessed. Avoid using words in your national language that would appear in a dictionary. Consider including a few digits in your passwords. Alternatively, let the system generate passwords for you automatically to improve the chances your passwords will not be guessed.
- Never write your password down anywhere.
- Never give your password to other users except under very unusual circumstances, and then be sure to change it immediately after the need for sharing has passed.
- Do not include your password in any file, including the body of an electronic mail message. (If anyone else reveals their password to you in this fashion, be sure to delete the information promptly.)
- Take special care when you find yourself about to log in on a previously turned on terminal. If so directed by your security manager, invoke the secure terminal server feature with the BREAK key.

- Change your password frequently. Password changes every 3 to 6 months are sufficiently frequent on most systems where there have been no password compromises and no sharing of passwords. DIGITAL recommends against sharing passwords; however, if passwords are shared, the frequency of password changes should be every month or two.
- Change your password immediately if you have any reason to suspect it may have been discovered. Report such incidents to your security manager, even at the risk of appearing paranoid.
- Avoid using the same password for your accounts on multiple systems, unless you have good reason to believe the practice is very safe.



ZK-2073-84

### 3.1.4 Account Expiration Times

When your account is created, the security manager may decide to specify a period of time after which the account will lapse. This is an important action if you will only need the account for a specific purpose for a limited time. For example, at universities the student accounts are typically authorized for a



single semester at a time. Security managers who implement account expiration times need not remember to disable expired accounts. Expired accounts will automatically deny logins at the desired time. In general, the use of account expiration times is a good practice under these kinds of circumstances.

Note that users receive no advance warning message prior to the expiration date, so it is important to know in advance what your account duration will be. The account expiration resides in the UAF record, which can only be accessed and displayed through the use of the VAX/VMS Authorize Utility by users with the SYSPRV privilege or equivalent—normally your system or security manager.

When your account does expire, you will simply receive an authorization failure message at your next attempted login. If you need an extension, you must follow the procedures defined at your site to request one. If your extension request receives approval, it is a simple matter for the security manager to use the VAX/VMS Authorize Utility to modify the account expiration field in your UAF record.

---

### 3.1.5 Causes of Login Failures

Sometimes you may attempt to log in and fail. There are a number of possible causes that this section describes. However, first you should rule out the more common causes from Table 3-2.

**Table 3-2 Common Symptoms and Causes of Login Failures**

Symptom	Cause
No response from the terminal	Attempting to use a defective terminal
No response from any terminal	Attempting to login when the system is down
Message: User authorization failure	Mistyping the username or password
Message: User authorization failure	Attempting to use an expired account
Message: User authorization failure	Attempting to use an expired password

Other reasons for login failures can be less obvious. Among the possibilities are the need for a system password, shift restrictions, prohibitions on certain kinds of logins, or system activation of evasion techniques for your terminal (to thwart illegal access attempts). The following sections will try to explain these causes and what, if anything, you can do about them.

#### **3.1.5.1 System Password Failures**

Your attempts to log in will fail if the terminal you attempt to use requires a system password and you are unaware of the requirement. As Section 3.1.3.1 explains, some systems require that a system password be entered from a particular terminal before anyone can login at that terminal. (This is an option the security manager may choose to implement.) To the uninitiated user, there is no telltale sign that a system password is required. There is no response, so the system appears to be down. All attempts at logging in will fail until the system password is entered. If you suspect that this is the problem, and you have not been given the password, you might try logging in at another terminal.

If you have been directed to use a terminal that requires a system password and have been told the password, simply perform the steps described in Section 3.1.3.1. If your attempts fail, it is also possible that someone changed the system password and forgot to notify you. In that case, you must move

to a different terminal that does not require a system password or else you must take steps to learn the new system password.

---

### 3.1.5.2 Login Class Restrictions

Other reasons why you may not succeed in logging in include attempting a class of login that is prohibited. For example, your security manager may have restricted you from logging in over the network. Possibly an unrestricted co-worker explains the procedure to you, so you attempt it, but you find you are unsuccessful. You receive a message telling you that you are not authorized to login from this source.

Your security manager has the power to restrict your logins to include or exclude any of the following classes discussed in Section 3.1.1: LOCAL, REMOTE, DIALUP, BATCH, or NETWORK. The general name INTERACTIVE is useful to include or exclude all three of the classes LOCAL, REMOTE, and DIALUP with one expression. Security managers normally explain all restrictions to their users, but it is possible that this detail was overlooked or perhaps you forgot what you were told.

---

### 3.1.5.3 Shift Restrictions

Another cause of login difficulty is failure to observe your shift restrictions. The security manager can restrict your logins to certain time periods within the day and/or certain days within the week. These restrictions are imposed on classes of logins. Sometimes the security manager applies the same work time restrictions to all classes of logins and sometimes the manager chooses to place different restrictions on different login classes. Naturally, the more finely tuned the hours and days become for the type of work being done, the more confusing it can be to remember the restrictions. However, if you attempt a login during one of your prohibited times for that login class, you fail with the notification that you are not authorized to login at this time.

Shift restrictions have additional implications beyond your interactive logins.

**Note:** Be aware that when shift restrictions apply to your batch jobs, you must not submit jobs that must begin to run outside your permitted work times. The jobs will not be run. Furthermore, the system will not automatically resubmit such jobs during your next available permitted

work time. Similarly, if you have initiated any kind of job and attempt to run it beyond your permitted time periods, the job controller will abort the uncompleted job when the end of your allocated work shift is reached. This job termination behavior applies to all jobs, not just batch jobs.

---

#### **3.1.5.4 Dialup Login Failures**

Your security manager can control the number of chances you are given to enter a correct password during a dialup login, before the connection is automatically broken. At some sites you will have only one chance, while other sites may be far more generous. Your security manager can tell you what your limit is.

If your login fails and you have some tries remaining, simply press the RETURN key and try again. You may do this repetitively until you succeed or you reach the limit. If the connection is lost, you can redial the access line and start all over again.

The typical reason for limiting the number of dialup login failures is to discourage users who are attempting to learn passwords with the trial-and-error technique. They already have the advantage of some anonymity because of the dialup line. If unlimited numbers of tries were permitted for a single dialup, they might be further encouraged to try this approach. However, limiting the number of tries per dialup will not necessarily stop this kind of *breakin attempt*. It simply requires the would-be perpetrator to redial and start another login.

---

#### **3.1.5.5 Breakin Evasion Has Been Activated**

If you or anyone else has previously made a number of failed attempts to login at the same terminal with your username, it is possible that the system is responding to its programming that says a breakin attempt is in progress. That is, the system concludes that someone is attempting to gain illegal access to the system using your username. As a result, the system has disabled even your valid logins on that terminal for a certain period of time to frustrate the would-be perpetrator (and you, too, if you forget that this possibility exists).

At the discretion of your security manager, *breakin evasion* measures may be in effect for all users of the system. These measures will work against you if you make numerous typing errors when trying to specify your password so that you exceed the number of failures allowed within a certain period of time. The security manager controls how many tries are allowed and the amount of time. Once breakin evasion tactics are triggered, you will be unable to log in to the terminal—even with your correct password—during a defined interval. Your security manager can tell you how long you must wait. Otherwise, if you have the option, you could move to another terminal to attempt a login.

Of course, if you have good reason to believe breakin evasion is preventing your login and you have not personally experienced any login failures, you should immediately reach your security manager. Together you should attempt another login, carefully checking the message that reveals the number of login failures since the last login, to confirm or deny your suspicion of breakin attempts. (If your system does not normally display the login message, your security manager can use the VAX/VMS Authorize Utility to examine the data in your UAF.) With prompt action, your security manager may locate the culprit attempting logins at another terminal.

---

### 3.1.5.6 Summary of Login Failure Sources

If you find you cannot login, and you have eliminated the common sources of login failures that Table 3-2 identifies, you should proceed to Table 3-3. Table 3-3 summarizes all the other possible symptoms and their sources that this section presents. By consulting these tables, you can avoid falsely alarming your security manager that equipment is defective or that someone has changed your password or deleted your account. On the other hand, once you are satisfied that none of the situations defined by either Table 3-2 or Table 3-3 is causing difficulty, by all means, seek assistance from your security manager or system manager.

**Table 3-3 Other Symptoms and Causes of Login Failures**

Symptom	Cause
No response from terminal	The device requires a system password.
No response from terminal to your entry of system password	The system password changed and you were not notified.
Message: Not authorized to login from this source	The attempted class of log in (LOCAL, DIALUP, REMOTE, INTERACTIVE, BATCH, or NETWORK) is prohibited.
Message: Not authorized to log in at this time	The day of the week or hours of the day are not permitted for you for this class of login.
Message: User authorization failure (and no known user failure occurred)	An apparent break-in has been attempted at the terminal using your username, and the system has temporarily disabled all logins at that terminal by your username.

### 3.2 Network Security Considerations for Users

This section describes several ways in which users can help make network accesses more secure. Among the topics described are access control strings in file specifications and command procedures, proxy logins, and proper use of the VAX/VMS Mail Utility.

#### 3.2.1 Network Access Control Strings

Network access control strings are designed to be included in the file specifications of DCL commands that work over the DECnet-VAX network. They permit a user on a local node to request an operation using a file on a remote node. They consist of the remote node name, the username for the remote account, and the user's password. Because they include sufficient information to permit anyone to break in to the remote account, they create a very serious security exposure.

You should do everything possible to protect access control string information. You should diligently avoid leaving the information revealed on either hardcopy or video terminals. Furthermore, you should avoid placing networking commands in command procedures where they would be likely targets for

discovery. The syntax that requires the username and password to be placed within quotation marks and followed by two colons makes searches for passwords in insufficiently protected files quite trivial. Whenever you find the three-character access control string terminator ("::), you can be pretty certain that a password immediately precedes it. If you absolutely must put networking commands that include access control strings in your command procedures, be certain to provide these files with optimum file protection, using the techniques that Chapter 4 describes. Better yet, investigate with your security manager the possibility of using proxy login accounts—so that you need not use access control strings. Section 3.2.2 explains more about this technique.

---

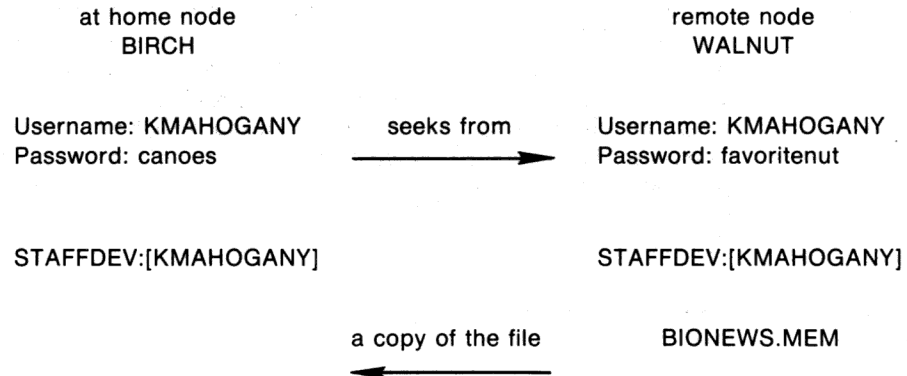
### 3.2.2 Proxy Logins

As described in Section 3.1.1.4, proxy logins are primarily a means of permitting users to perform file accesses across a network without specifying user names and passwords in an access control string that is susceptible to interception, observation, or other forms of discovery. While proxy logins appear very simple and straightforward from the user's point of view, the security benefits are derived at the expense of the security manager's time and expertise.

Before a user can issue a request that initiates a proxy login, the system or security manager at the remote node must create a proxy account for the user. Proxy accounts are created with the VAX/VMS Authorize Utility, in much the same way that a regular user account is created. However, proxy accounts require that a *network user authorization file (NETUAF)* be created. The NETUAF records identify each remote user who is to be allowed access to a proxy account.

The following examples illustrate the differences the user sees between a normal network login request and a proxy login request. In the first case, the user has two user accounts, one on the node BIRCH with the user password "canoes" and one on the node WALNUT with the user password "favoritenut". The user has logged into BIRCH and wants to copy the file BIONEWS.MEM over from the default device and directory of the account on the node WALNUT. The situation is diagrammed in Figure 3-2.

**Figure 3-2 Illustration of File Sharing Over a Network**



ZK-2036-84

The user issues the following command:

```
$ COPY WALNUT"KMAHOGANY FAVORITENUT"::BIONEWS.MEM BIONEWS.MEM
```

Notice that since the password of the KMAHOGANY account on WALNUT echoes, it is revealed to any passerby who observes the screen during its display. If Kay Mahogany uses a hardcopy terminal and does not properly dispose of the output, the password is also exposed, but to a potentially wider audience. With a video terminal, if Kay fails to log out and leaves her terminal unattended, even if she takes the precaution of clearing the screen, her password information is also subject to review through the DCL commands CTRL/B or RECALL /ALL.

In the next example, the security manager at the node WALNUT also maintains an account for Kay Mahogany with the username KMAHOGANY and the default device and directory of STAFFDEV:[KMAHOGANY]. However, this time the security manager at WALNUT authorizes user BIRCH::KMAHOGANY to perform proxy logins into the WALNUT::KMAHOGANY account. As the owner of the account on WALNUT, Kay has a password, but she will not need to specify it in her access control string when she performs network commands from BIRCH, because the



## Security for the User

system will perform a proxy login into her account. (Indeed, this is deliberate, to protect the account.) To copy the file BIONEWS.MEM from the default device and directory of the KMAHOGANY account on WALNUT, Kay Mahogany issues the following COPY command:

```
$ COPY WALNUT::BIONEWS.MEM  BIONEWS.MEM
```

Here the network does the internal verification that the incoming user is BIRCH::KMAHOGANY and a network login is performed on the node WALNUT so that the user KMAHOGANY receives proxy access to her files in the WALNUT account. There is no exchange of passwords.

This is an example of a single user proxy login account; Kay is the only remote user allowed to access the account. However, it is also common practice to authorize groups of users from foreign nodes to share in the use of single home node accounts, as in the next example.

Suppose the security manager at the node WALNUT creates a general access account with the username GENACCESS that permits only network logins. The default device and directory are STAFFDEV:[BIOSTAFF]. Next the security manager authorizes a number of remote users for proxy logins to this account and includes Kay Mahogany on node BIRCH as one of those authorized users. The owner of the general access account on WALNUT has a password, but none of the remote users like Kay Mahogany need to know it. (Again, this is deliberate, to protect the account.) To copy the file BIONEWS.MEM from her own directory on the default device accessed through the GENACCESS account on WALNUT, Kay Mahogany issues the following COPY command:

```
$ COPY WALNUT::[KMAHOGANY]BIONEWS.MEM  BIONEWS.MEM
```

Notice that this command specifies an explicit directory for node WALNUT. This is only necessary because the file BIONEWS.MEM resides in STAFFDEV:[KMAHOGANY] while STAFFDEV:[BIOSTAFF] is the default device and directory for the proxy login account GENACCESS on node WALNUT. Note that the protection for the file BIONEWS.MEM must permit access to the GENACCESS account, or the example fails.

Observe how this arrangement also succeeds in eliminating the need to forward the password as a string. This is the key characteristic of proxy logins, whether the accounts are set up for single remote users or groups of remote users. Had the file BIONEWS.MEM been moved to the directory [BIOSTAFF], the two copy commands illustrating proxy logins could have been identical.

Thus, while proxy logins require more setup effort on the part of the managers at the nodes, they provide more secure network access and involve much simpler procedures for the users.

---

### 3.2.3 Using the VAX/VMS Mail Utility

When you use the VAX/VMS Mail Utility to send electronic mail, you should apply discretion both to your content and the selection of your audience. Keep security considerations uppermost. You should never reveal your password or send details regarding how to use your account through the mail. You have no control on how well the recipients will protect the information you send. While it is true that the default file protection applied by the VAX/VMS Mail Utility helps to discourage mail browsers, it is far from tamperproof. The recipient (or anyone with sufficient privileges) can quickly change the protection on mail files. Remember that mail files always make interesting targets during security assaults.

As the recipient of electronic mail, you should promptly read and properly dispose of your messages. Sensitive material demands special care. Hardcopies should be handled according to company guidelines. Files that you plan to retain may require additional file protection. Chapter 4 describes file protection in detail.

---

### 3.3 Logging Off the System

First of all, some criteria must be established for when to log out of the system. It is very tempting to leave your terminal on line while you go on a coffee break or pick up your mail. After all, you expect to return shortly and you might like to avoid logging out and then logging in again. Occasionally the temptation stems from the fact that you just started a long job and you will be unable to use your terminal until the job completes.

The thought of users walking away from their terminals while still logged in causes recurring nightmares for security managers. It represents one of the greatest sources of break-ins from the inside. Here you may have a system with elaborate passwords, very restricted privileges, good file protection designs, and every appropriate security feature correctly implemented. But, in one brief lapse, one user can undermine the whole painstaking process.



ZK-2065-84

When you leave your terminal on line and your office open, you have effectively given away your password, given away your privileges and left your files and those of the other members of your group unprotected! Any user who walks up to your terminal can transfer all of your files and all of those belonging to members of your group that you have READ access to (unless they are further protected by access control lists). The transfer of many files could be done in a matter of seconds! And you might not be the wiser if you return and find a logout message as if you had logged out. You might shake your head for a minute, and wonder if you really did log out and forgot that you had, but you might also think your security manager caught you, and be too embarrassed to mention it. Just think about that for a minute.

Of course, the malicious insider may take this opportunity to go on a rampage and rename or delete your files instead. (Chapter 4 reveals how the malicious user can use your account to rename the files of any user whose parent directory permits you WRITE access.) Worse yet, all files that you have WRITE access to can be deleted, and in even less time than it would take to transfer them. This set of files includes those belonging to members of your group and those where the fact that you hold a valid identifier for the file's access control list allows you to gain the access. If you have any special privileges, particularly the SETPRV privilege, the havoc can be exorbitant.

Consider how much work might have been done since the last incremental backup with the VAX/VMS Backup Utility, and you will sense how much might be lost. Sometimes it helps to express the work in person-days. Suppose there are 20 people in your group including yourself, all working a normal eight-hour workday and all actively working on files. The incremental backups are done every evening at midnight. Leaving your terminal exposed to this type of catastrophe for a 3:00 p.m. coffee break could result in the loss of 100 work hours—or 12.5 person-days.

Unfortunately, in losses of this kind, there are generally even greater losses, because of the involvement of additional personnel in restoring the files from the backup copies and the emotional upsets of the creative individuals whose work has been lost. Recoveries, when possible, are usually less than satisfactory. This type of illustration should help emphasize the importance of never leaving your terminal unattended while you are logged in.

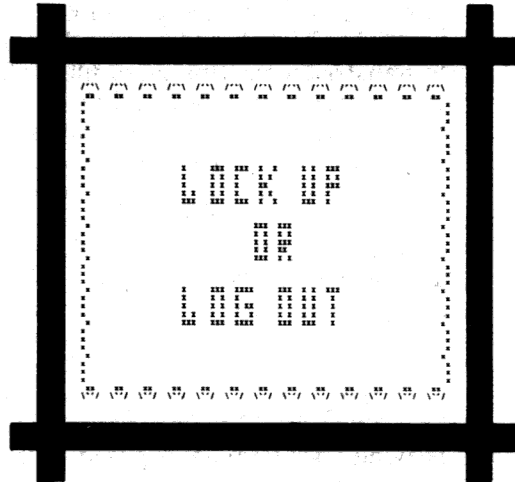
When you leave your terminal logged in and otherwise unprotected, you may also reveal the work that you have been doing. This is a relatively minor concern when compared to the other damage that can result when terminals are left exposed and logged in, but it is worth considering. Even if you clear the screen of a video terminal before you leave, another user can walk up to the terminal and issue one of the DCL commands RECALL/ALL or CTRL/B. Your previous 16 or 20 commands will be displayed.

Wisdom dictates that you really have just two choices when you are called away from your area while you are logged in:

- 1 Lock your area if you can

## 2 Log out

You may find situations where both actions are warranted.



ZK-2071-84

The proper procedures for logging out vary depending on whether the terminal is a hardcopy or a video terminal. The next two sections address these two procedures separately.

### 3.3.1

#### Logging Out with Video Terminals

There are several steps you may want to consider each time you plan to log out. Security conscious sites will have very few hardcopy terminals because of the security breaches possible if the output falls into the wrong hands. So, this discussion will focus on the concerns for logging out with a video terminal.

First, how much information can you afford to leave on the screen at logout time? At low-level security sites, there is seldom concern with what has been left on the screen, so you may be directed to simply log out. However, at sites with medium-level security concerns it may be desirable to leave nothing but the logout message on your screen. If your terminal is in the VT100 series of terminals, you can accomplish this if you first use the setup feature of your terminal to clear the screen. (You press the SET-UP key, then the key marked for reset (the 0 key), followed by the RETURN key.) For the

## Security for the User

VT200 series of terminals, you can accomplish this by pressing the SET-UP key, then selecting the item from the resulting menu that corresponds to CLEAR DISPLAY.

Once the screen has cleared, and the DCL prompt returns, you can enter the DCL command LOGOUT. Since your screen is clear, the cursor is positioned at the top of the screen and the only information remaining will be your logout command and the logout completion message in the following form:

```
$ LOGOUT
RDOGWOOD      logged out at 14-AUG-1985 19:39:01.43
```

At high-security sites it is common practice to turn off your video terminal every time you log out. Just as there are reasons for security conscious sites to avoid displaying the name of the node and the operating system name and version at login, there are reasons to avoid leaving logout messages on display at video terminals at logout time. After all, the logout message reveals a currently active username. In cases where users log off after a remote login, the name of the node they return to after the remote logout is also revealed. Note also, that in the case where a user has accessed multiple accounts remotely over the network, the final sequence of logout commands reveals all the nodes and the usernames that are accessible to the user on each node, with the exception of the name of the furthest node reached. To those who can recognize the operating system from the prompt or a logout message, this small amount of information will also reveal the operating system.

Remember, if the system fails before you logout, there may be some potentially valuable information left on your screen for an insider with bad intentions. The best procedure (if you plan to wait in the immediate vicinity) is to turn your video terminal first off and then on. Next you wait for the system initialization message or your first chance to reaccess the terminal concentrator. If you plan to leave the area, do not turn the terminal on again until you return. (Observe the precautions described in Section 3.1.3.8 to avoid the password grabbers.)

The issues involved here are not preserving the equipment or reducing electricity consumption, but maintaining security.

---

### 3.3.2 Logging Out with Hardcopy Terminals

When you need to log out from a hardcopy terminal, you simply issue the DCL command LOGOUT. If you have performed some remote logins, you must enter successive LOGOUT commands until you no longer are returned to a node and a DCL prompt. Your primary concern is to promptly and properly remove, file, or dispose of all hardcopy output that might reveal security information to a probing insider. Your security manager should provide direction on the preferred procedures. Many sites employ paper shredders or locked receptacles for this purpose. Handle any of the output that you plan to save just as carefully.

If the system fails before you can log out, dispose of the hardcopy output, as outlined above. You should turn your terminal off if you will not be waiting in the immediate vicinity for the system to be initialized.

---

### 3.3.3 Logging Out with Disconnected Processes

The next series of suggestions concern logging out when you know that you have other disconnected processes. These suggestions are not related to security. Rather, they are good procedures that conserve system resources.

- Issue the DCL command SHOW USERS to determine if you have other disconnected jobs.
- Issue the DCL command CONNECT/CONTINUE to log out of the current process and connect back through each of the associated virtual terminals (as noted by the terminal prefix of VTA), until you have reached the last existing process.
- Issue the DCL command LOGOUT

If you do not perform these steps, the system will normally remove your disconnected processes after a certain interval, anyway. If your security manager has enabled a long interval, the procedure above is more valuable than if the interval is relatively short. You should be guided in this by your security manager.



---

### 3.3.4 Logging Out from a Dialup Login

Your security manager may request that you always use the /HANGUP qualifier with your final LOGOUT command when you log out from a dialup login and anticipate no further immediate use of the line. This is especially warranted if the dialup line you use is in a public area or there might be an unsupervised opportunity for someone to use that terminal after you. The /HANGUP qualifier will work for all terminals that have been set up with the /HANGUP terminal characteristic through the DCL command SET TERMINAL /PERMANENT/HANGUP. The use of the /HANGUP qualifier on a LOGOUT command directs VAX/VMS to automatically break the connection to the dialup line after the logout. Thus, you are ensured that no one else can take advantage of an open access line; they must know the access number and redial for themselves.

As a final note, since access lines are a limited resource, this practice makes good resource management sense.

---

## 3.4 Summary of Good User Practices

This chapter has presented many suggestions for users to help maintain a secure system. Although many security features are implemented by the security manager as requirements for all users, there remain a number of ways that users can contribute to system security on their own. The list below reviews voluntary actions that you should keep in mind and practice where appropriate. Included with this list are cross-references to parts of this chapter where each topic is initially discussed.

- Protect your password—Section 3.1.3.10
- Check your last login messages each time you log in and report any inexplicable messages to your security manager—Section 3.1.2.4
- Lock up and/or log out when you need to leave your terminal and/or area—Section 3.3
- Use the /HANGUP qualifier on your final LOGOUT from a dialup line—Section 3.3.4
- Properly dispose of hardcopy output from your terminal—Section 3.3.2

## Security for the User

- Turn off your video terminal to erase revealing displays—Section 3.3.1
- Use proxy logins wherever possible—Section 3.1.1.4

If you follow the procedures requested by your security manager and for good measure include any of the above suggestions that were omitted, you will do much to make your system secure. There is more potential gain for you than you may immediately realize. A closer look reveals that inadequate security measures will affect all users.

When unwanted outsiders gain access to systems, the least harmful thing they can do is consume resources. In that case, performance degrades unnecessarily for all users who need to use the system. Among the more damaging things outsiders or insiders can do that could affect you personally is destroying your files. Also, when a company's profits are reduced (possibly through embezzlement or lost contracts resulting from security breaches), the damage soon takes its toll on raises, benefits, hiring, planned improvements, and other aspects of the working conditions. Another extreme is the potential loss of jobs for yourself or your friends when your organization loses its competitive advantage from the escape of proprietary information. Remember, too, that when users fail to cooperate, security managers are forced to adopt stronger and stronger measures to protect their companies and their resources. With each new control imposed, the system becomes unfriendlier, slower, and harder to use—for everyone.

You will be doing yourself, your coworkers, and your company a great service when you recognize the importance of security and do your part to promote it.

# 4

## File Protection Features

---

Chapter 3 discusses a number of security-related topics of interest to most users. It does not discuss file protection. Most users will understand the basics of file protection from reading the *VAX/VMS DCL Dictionary*. Many users will not require additional discussion. However, the file protection mechanisms are extremely important tools for enhancing system security that should not be neglected. Sites that are concerned about system security will undoubtedly need to understand the mechanisms and apply them appropriately. This chapter tries to assist all readers in gaining a better understanding of the mechanisms and their use.

File protection techniques require a little work to master and use consistently, but they are worth the effort. VAX/VMS offers two primary protection mechanisms. The first is the standard protection based on the user identification code (UIC), known as *standard UIC-based protection*. This protection mechanism controls access according to the user categories of SYSTEM, OWNER, GROUP and WORLD. The second protection mechanism relies on *Access Control Lists*, also known as *ACLs*.

All users will encounter the standard file protection mechanism. Some users will also find themselves using the access control lists. Access control lists are important file protection tools available to all VAX/VMS users and generally adopted at sites with medium to high requirements for system security. ACLs are also important in environments with complex patterns of file sharing. As security requirements increase, so will the use of ACLs. The next few sections provide more details about each file protection mechanism.

---

## 4.1 How the System Determines Access

There is an interaction between ACLs, the UIC-based protection code, and user privileges that determines the outcome every time a user requests access to an object. This section introduces, in a simplified way, the order in which VAX/VMS evaluates each of these possible components, so that you can understand the importance of each in resolving a request for access.

VAX/VMS follows these steps to determine if a user is allowed access to a particular object:

- 1 If an object has an associated ACL, the system uses the ACL to determine whether the user gains access to the object. If the ACL specifically grants the requested access to the user, the access is given and all further testing stops. However, if the ACL does not explicitly allow or refuse the user access, then the system uses UIC-based protection to determine access. If the ACL denies access, the system uses only the SYSTEM and OWNER fields of the UIC-based protection to further decide whether the user should gain access.
- 2 If an object does not have an associated ACL, the system uses UIC-based protection to determine access. Access is granted or denied, based on the relationship between the object's UIC and the user's UIC. (See Section 4.2.3.)
- 3 The GRPPRV, SYSPRV, READALL, and BYPASS privileges amplify the privilege holder's access to objects under certain circumstances. (See Section 4.2.5.)

The following sections discuss each of these components in detail. Section 4.6 concludes the detailed discussions by presenting a summary flowchart that depicts this evaluation process in greater detail. For now it is sufficient to recognize four key facts:

- ACLs are always evaluated first.
- When an ACL fails to specifically grant access, UIC-based protection is checked.
- When an ACL specifically denies access, the user still may acquire access by belonging to the SYSTEM or OWNER categories and being eligible for access through them or by possessing privileges.

- Users who possess certain system privileges may be entitled to access regardless of the protection offered by the ACLs or the protection code.

---

### 4.2 Standard UIC-based Protection

When security managers create accounts, they take two actions that affect the UIC-based protection:

- They establish each account with a standard default protection code for all files the user creates in the initial top-level directory.
- They assign each user as a member of a group.

It may be that your default protection code and group assignment are sufficient for all the files that you will work with and you will rarely find yourself changing the file protection with the DCL command SET PROTECTION. However, it is more typical that you will need to change the protection on certain files and to understand the mechanism more thoroughly.

Each user of the system has a UIC defined in the system UAF. Each system object also has an associated UIC, defined to be the UIC of its owner, and a protection code that defines who is allowed what type of access. The relationship between the UIC of the user and the UIC of the object controls access to the object.

---

#### 4.2.1 UICs and Protection

UIC-based protection is determined by an owner UIC and a protection code. UIC-based protection controls access to objects such as files, directories, and volumes. A volume is the entity that exists when a medium is mounted on a device. For example, disk packs and reels of magnetic tape are called volumes when they are mounted on disk and magnetic tape drives, respectively. For disk volumes, the system provides protection at the file, directory, and volume levels. For magnetic tape volumes, the system provides protection only at the volume level.

Thus, in addition to protecting the files on mounted disk volumes, the system provides overall volume protection for disks and magnetic tapes. This volume protection is coded into the home block of the disk or magnetic tape. For more information about setting volume protection characteristics for disks and magnetic tapes, see the *Guide to VAX/VMS Disk and Magnetic Tape Operations*. You might also consult the descriptions of the DCL commands INITIALIZE, MOUNT, and SET VOLUME in the *VAX/VMS DCL Dictionary*.

VAX/VMS also provides protection for record-oriented devices such as terminals, line printers, mailboxes, and special-purpose devices. See the description of the SET PROTECTION/DEVICE command in the *VAX/VMS DCL Dictionary* for information on how to apply protection to record-oriented devices.

---

### 4.2.2 Specifying UICs

A UIC has two formats: numeric and alphanumeric. When a DCL command requires a UIC specification, you can use either format.

---

#### 4.2.2.1 Numeric Format UICs

A UIC in numeric format contains a group number and a member number in the format:

[group,member]

The brackets are required in the UIC specification. The group number is an octal number in the range of 1 through 37776; the member number is an octal number in the range of 0 through 17776. You can omit leading zeros when you are specifying group and member numbers.

---

#### 4.2.2.2 Alphanumeric Format UICs

A UIC in alphanumeric format consists of a member name and, optionally, a group name in the format:

[member]

[group,member]

The brackets are required in the UIC specification. The group and member names can each contain up to 31 alphanumeric characters and must contain at least one alphabetic character. The names can include the characters A through Z, dollar signs (\$) and underscores (\_), and the numbers 0 through 9.

---

#### 4.2.2.3 UIC Translation and Storage

Regardless of the format you use, the system translates a UIC to a 32-bit value that represents a group number and a member number; the high-order 16 bits contain the group number and the low-order 16 bits contain the member number. When presented with the problem of translating an alphanumeric UIC such as [J\_JONES], VAX/VMS equates the member part of the alphanumeric UIC to both the group and member parts of a numeric UIC. The resulting 32-bit numeric UIC is kept in the *system rights database*, which is a file containing information pertaining to the access rights and attributes associated with identifiers and the holders of those identifiers.

This method of storing alphanumeric UICs dictates that member names must be unique and that no member can participate in more than one group. That is, each member name must be unique for each user on the system. For example, you could not have the two UICs [GROUP1,JONES] and [GROUP2,JONES] on the same system because the member JONES can have only one associated numeric UIC.

A group name is associated with the group portion of a UIC. When the system translates an alphanumeric UIC that includes both a group and a member name, the system obtains the longword integer associated with the member and checks the group name against the member.

The system manager assigns a UIC to each user with the VAX/VMS Authorize Utility. Since an alphanumeric UIC is equated to a numeric UIC in the rights database, you can generally specify either format to refer to a user.

For example, the following UIC specifications could all be valid for the user JONES:

[360,031]  
[JONES]  
[GROUP1, JONES]

### 4.2.3 How UIC-based Protection Controls Access

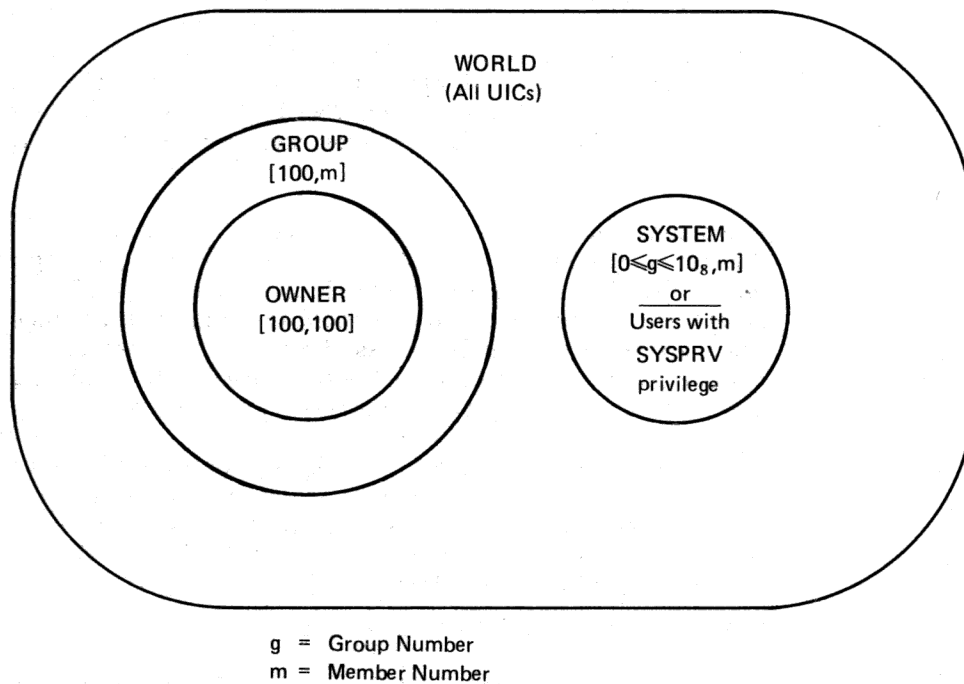
As indicated in Section 4.1, when a user attempts to gain access to an object (such as a file or volume), in almost all cases the system resorts to checking the UIC-based protection code. This involves comparing the user's UIC to the owner UIC of the object. (The only exception occurs when there is an ACL on the object that grants access immediately to the requesting user.) To understand the role of the protection code and how it directs the outcome of a request for access, you must first understand that the user always falls into one or more of the following categories, once the two UICs are compared:

SYSTEM	1. All users who have system privilege (SYSPRV).
	2. Users with low group numbers, usually from 1 through 10 (octal). However, the exact range of system group numbers is determined by the system manager (with the SYSGEN parameter MAXSYSGROUP) when the system is generated, and may range as high as 37776 (octal). These group numbers are generally for system managers, security managers, system programmers, and operators.
	3. Users with the user privilege GRPPRV whose UIC group matches the group of the object's owner.
	4. For files on disk volumes, users whose UIC matches the owner UIC of the volume on which the file is located.
OWNER	The user with the same UIC as the user who created and therefore owns the object.
GROUP	All users, including the owner, who have the same group number in their UICs as the object's owner.
WORLD	All users, including those in the first three categories.

Figure 4-1 illustrates the relationships of these categories to each other.



**Figure 4-1 Illustrating User Categories with a UIC of [100,100]**



NOTE: THE SYSTEM MANAGER CAN EXTEND THE SYSTEM GROUP NUMBER LIMIT TO 377<sub>8</sub>

ZK-778-82

Through the protection code, each of the categories of user can be allowed or denied any of the following types of access:

- READ
- WRITE
- EXECUTE
- DELETE

## File Protection Features

The previous list omits *CONTROL access* since it is never specified in the standard UIC-based protection code. However, CONTROL access is a fifth type of access that can be specified in an ACL and is automatically granted to certain user categories when UIC-based protection is evaluated. CONTROL access grants the accessor all the privileges of the object's actual owner. For example, the user who acquires CONTROL access can change the protection and file characteristics, just as the owner could. Thus, it is inherent that users in the SYSTEM or OWNER categories always have CONTROL access, while users in the GROUP or WORLD categories never receive CONTROL access.

The actual abilities conveyed by the access types READ, WRITE, EXECUTE, DELETE, and CONTROL vary depending on the situation where they apply. For example, EXECUTE access permits very different operations depending on whether it is granted for general file access, directory file access, or volume access. The sections that describe each of these objects also detail the abilities that each access type provides.

The protection code describes the categories of users who have access to an object, and the type of access that each category has. For example, the protection code in the next example specifies that users in the SYSTEM and OWNER categories have READ, WRITE, EXECUTE, and DELETE access:

```
SYSTEM:RWED, OWNER:RWED, GROUP:RE, WORLD:RE
```

In this case, users in the GROUP and WORLD categories have only READ and EXECUTE access.

---

### 4.2.4 Protection Code Syntax

The following syntax rules apply to protection codes:

- When you specify a protection code, you must abbreviate access types to one character: R, W, E, or D. User categories can be entered in full or truncated to any number of characters. Separate each user category from its access types with a colon. If you specify more than one user category, separate the categories with commas and enclose the entire code in parentheses. For example, the following DCL command that sets the protection code illustrates these rules:

```
$ SET PROTECTION=(S:RWED,OWN:RWED,GROUP:R,W:R) DATAFILE.DAT
```

- You can specify the user categories and access types in any order. If you omit an access type for a user category, that category of user is denied that type of access. If you want to deny all access to a user category, specify the user category but do not list any access types. Also, be certain to omit the colon after the user category when you are denying access to a category of users. For example, the following DCL command sets the protection code to deny WRITE and DELETE access to the GROUP category and denies all access to the WORLD category:

```
$ SET PROTECTION=(S:RWED,O:RWED,G:RE,W) DATAFILE.DAT
```

- When you omit a user category from a protection code, the current access allowed that category of user remains unchanged. For example, assume the protection code in the immediately preceding example is in effect. The following DCL command resets the protection on the file DATAFILE.DAT so that the SYSTEM and OWNER categories of users can no longer DELETE the file, but the GROUP category will still gain READ and EXECUTE access and the WORLD category will still be denied any access:

```
$ SET PROTECTION=(S:RWE,O:RWE) DATAFILE.DAT
```

- When you set the protection for **magnetic tape** volumes, regardless of the specification in the protection code, the SYSTEM and OWNER categories always have access.

---

### 4.2.5 How Privileges Affect Protection

The VAX/VMS system features *privileges* that system and security managers can assign to users when they create or modify the user's accounts. Four of the system privileges can affect the access a user actually receives, regardless of the access apparently dictated by either an ACL that may be present for the object or the access granted through matching the user's category in the protection code. These four privileges are:

- SYSPRV
- GRPPRV
- READALL
- BYPASS

## File Protection Features

If a user holds any one of these privileges, the outcome of the protection check may be quite different from your initial expectation.

<b>SYSPRV</b>	A user with SYSPRV privilege will receive the access accorded to users in the SYSTEM category.
<b>GRPPRV</b>	A user with GRPPRV privilege, whose UIC group matches the group of the owner of the object, receives the same access accorded to users in the SYSTEM category. Thus, the user with GRPPRV privilege is able to manage a group's files.
<b>BYPASS</b>	A user with BYPASS privilege receives all types of access to the object, regardless of its protection.
<b>READALL</b>	A user with READALL privilege receives READ and CONTROL access to the object, even if that access is denied by the ACL or UIC-based protection. In addition, the user may receive any other access that is granted through the protection code.

Thus, whenever you define ACLs or protection codes for your objects, you should remember that users with these privileges are entitled to special access to objects throughout the system. For example, there is no way you could stop a user with the BYPASS privilege from accessing your files. The ultimate protection of your objects depends on the judgement of your security manager in granting these privileges and the practices at your site that prevent the wrong users from subsequently acquiring the privileges. To some extent, users must take this protection for granted and focus their attention on providing proper protection from users who lack these privileges.

---

### 4.2.6 How the System Interprets a Protection Code

To determine access to an object (such as a file or a device), the system uses the object's protection code for each user category. The system checks user categories one-by-one, in the following sequence:

- 1 OWNER
- 2 WORLD
- 3 GROUP
- 4 SYSTEM

You can access a resource as soon as the system finds a user category that you fit into that gives you the access you have requested. In order to deny access to a user category, be sure to deny access to all the outermost categories.

For example, you might initially think that the following protection code denies DELETE access to the OWNER category:

```
SYSTEM:RWED, OWNER:RW, GROUP:RW, WORLD:RWED
```

However, the owner of the file will still be able to delete the file. Even though DELETE access is denied through the OWNER category, the system continues checking the remaining categories for permission to grant access. Because the owner also fits in the WORLD category (which applies to all users) and the WORLD category is permitted DELETE access, the system grants DELETE access to the owner.

---

### 4.2.7 How the System Interprets the Access Types for Objects

Depending on the object, VAX/VMS applies different meanings to the access types READ, WRITE, EXECUTE, DELETE, and CONTROL. This section discusses various objects and the different interpretations of access types for each.

---

#### 4.2.7.1 Access to Disk Files

Each file on a disk has its own protection code. When applied to files, access types have the following meanings:

READ	The right to examine (read), print, or copy the file
WRITE	The right to write to or modify the file
EXECUTE	The right to execute a file that contains an executable program image or DCL command procedure
DELETE	The right to delete the file
CONTROL	The right to change the protection and file characteristics of the file

Note that READ access also implies EXECUTE access. Also note that WRITE access permits a user to change the contents of a file, possibly deleting sufficient portions of it to render it useless, even though the file cannot be removed from the directory without DELETE access.

**Note:** To open a file for writing, you must have both READ and WRITE access; VAX/VMS does not support write-only files.

### 4.2.7.2 Access to Directory Files

Each *directory file* (easily recognized as a file with the file type DIR) has a protection associated with it. This directory protection can override the protection of individual files in the directory. When applied to directories, access types have the following meanings:

READ	The right to examine (read) or list the directory file
WRITE	The right to write to or modify the directory file
EXECUTE	The right to look up files in the directory if you explicitly specify the file name
DELETE	The right to delete the directory file
CONTROL	The right to change the protection and file characteristics of the directory file

Note that READ access implies EXECUTE access.

If you have READ access to a directory file, you can display the contents of the directory file with the DCL command DIRECTORY. You can use wildcards (explicitly or implicitly). For example, if you have READ access to the directory [MALCOLM], you can obtain a listing of all the files contained in the [MALCOLM] directory by issuing the following command that employs implicit wildcards:

```
$ DIRECTORY [MALCOLM]
```

In addition, you can access any file stored in the directory unless the protection on that file denies you access. However, if a directory denies you READ access, you cannot look up or access even those files in the directory that permit access to users in your group. Strictly speaking, the Files-11 structure is not hierarchical, and it is possible to access files without using the directory in which they are listed, through suitable programming. Therefore, to guarantee protection, you should also protect individual files.

WRITE access allows you to write to the directory file. You must have both READ and WRITE access to a directory in order to create files in that directory, to rename files, or to perform any file operation that involves changes to the directory file.

## File Protection Features

EXECUTE access has a special meaning when it is applied to directories. EXECUTE access allows you to use the DIRECTORY command to look up files that you can identify by name. In addition, you can access files in the directory that are not protected against users in your category—if you do not perform an operation that modifies the directory file. However, you cannot list all the entries in the directory by using wildcards.

In the next example, assume that you have EXECUTE access to the [MALCOLM] directory and you issue a DIRECTORY command without naming any files. The system responds with an error message and does not list the files in the [MALCOLM] directory:

```
$ DIRECTORY [MALCOLM]
```

However, if you know that the file DATAFILE.DAT exists in the [MALCOLM] directory, you can issue a more specific command, as in the following example, and the system will list the corresponding directory information.

```
$ DIRECTORY [MALCOLM]DATAFILE.DAT;0
```

Therefore, EXECUTE access provides some, but not all, of the operations that READ provides.

DELETE access allows you to delete a directory file. You must remove all entries from a directory file before you can delete it. When you create a directory file with the CREATE /DIRECTORY file you do not, by default, receive DELETE access. If you want to delete a directory file, you must use the SET PROTECTION command to explicitly assign DELETE access to the OWNER category.

To protect your files, you must ensure that they are adequately protected at both the directory and file level.

---

#### 4.2.7.3 Access to Volumes

When applied to volumes, access types have the following meanings:

READ	The right to examine, print, or copy files on a volume
WRITE	The right to modify or to write existing files on a volume
EXECUTE	The right to create files on the volume and to write into them
DELETE	The right to delete files on the volume
CONTROL	The right to change the protection and ownership of the volume

Note that READ access on volumes limits the access to read only.

**Note:** EXECUTE and DELETE access are not valid for magnetic tapes. Granting a category of users WRITE access to a tape volume automatically permits them to have READ access to the volume.

---

#### 4.2.7.4 Access to Global Sections

When applied to global sections, access types have the following meanings:

READ	The right to map the section for read access
WRITE	The right to map the section for write access
EXECUTE	The right to map the section for execute access (available only to privileged software)
CONTROL	The right to change the access control list (applies only to PFN and page file global sections)

---

#### 4.2.7.5 Access to Devices

When applied to devices, access types have the following meanings:

READ	The right to issue read requests to the device
WRITE	The right to issue write requests to the device
CONTROL	The right to change the device ACL



---

#### 4.2.7.6 Access to Logical Name Tables

When applied to logical name tables, access types have the following meanings:

READ	The right to look up logical names in the table
WRITE	The right to create and delete logical names in the table
DELETE	The right to delete the table
CONTROL	The right to change the logical name table ACL

---

#### 4.2.7.7 Access to Queues

Operations that apply to a queue or to specific jobs in a queue are controlled by UIC-based protection in the same way access to other system objects is controlled. The ability to control queue operations through UIC-based protection allows you to restrict the types of jobs and users for a particular queue.

When you initialize a queue, the queue is assigned an owner UIC and a protection mask. Jobs are assigned an owner UIC equal to the UIC of the process that submitted the job. Each operation that is performed on a queue or a job in a queue is checked against the owner UIC, protection of the queue and the job, and the privileges of the requestor.

Operations that apply to jobs are checked against the READ and DELETE protection specified for the queue and the owner UIC of the job. In general, READ access to a job allows a user to see the attributes of a job, and DELETE access allows the user to delete the job.

Operations that apply to queues are checked against the WRITE and EXECUTE protection specified for the queue and the owner UIC of the queue. A user with WRITE access to a queue can submit jobs to that queue. Users with EXECUTE access to a queue may act as the operator for that queue with the ability to affect any jobs in the queue. Users with operator (OPER) privilege have EXECUTE access to all queues. OPER privilege also enables users to establish queues and affect accounting.

## File Protection Features

The following table summarizes the privileges required for various queue operations:

Command	Privilege Required
START/QUEUE/MANAGER	OPER and SYSNAM
STOP/QUEUE/MANAGER	OPER and SYSNAM
DEFINE/CHARACTERISTIC	OPER
DEFINE/FORM	OPER
DELETE/CHARACTERISTIC	OPER
DELETE/FORM	OPER
DELETE/QUEUE	OPER
INITIALIZE/QUEUE	OPER
SHOW QUEUE	OPER, EXECUTE access to the queue, or READ access to the job
SYNCHRONIZE	OPER, EXECUTE access to the queue, or READ access to the job
PRINT	OPER, EXECUTE access to the queue, or WRITE access to the job
SUBMIT	OPER, EXECUTE access to the queue, or WRITE access to the job
ASSIGN/MERGE	OPER or EXECUTE access to the queue
ASSIGN/QUEUE	OPER or EXECUTE access to the queue
DEASSIGN/QUEUE	OPER or EXECUTE access to the queue
SET QUEUE	OPER or EXECUTE access to the queue
START/QUEUE	OPER or EXECUTE access to the queue
STOP/QUEUE	OPER or EXECUTE access to the queue
STOP/QUEUE/NEXT	OPER or EXECUTE access to the queue
STOP/QUEUE/RESET	OPER or EXECUTE access to the queue
DELETE/ENTRY	OPER, EXECUTE access to the queue, or DELETE access to the job
SET QUEUE/ENTRY	OPER, EXECUTE access to the queue, or DELETE access to the job
STOP/QUEUE/ABORT	OPER, EXECUTE access to the queue, or DELETE access to the job
SET RESTART_VALUE	No privilege

---

## 4.2.8 Establishing and Changing UIC-Based Protection

This section describes how UIC-based protection is initially established for volumes, files, global sections, devices, logical name tables, and queues. It also describes how you can change the protection of volumes, files, devices, and queues.

---

### 4.2.8.1 Establishing and Changing Volume Protection

VAX/VMS determines the UIC-based volume protection when a volume is mounted; it can be defaulted from the protection recorded on the volume or it can be explicitly specified. (See the descriptions of the INITIALIZE and MOUNT commands in the *Guide to VAX/VMS Disk and Magnetic Tape Operations* for more information on setting volume protection when you mount a volume.) To change the protection of a disk volume, use the SET VOLUME command.

Volume protection for magnetic tape volumes differs significantly from disk volume protection. The protection applied to a **magnetic tape volume** applies equally to all files on the volume. VAX/VMS applies only READ and WRITE access restrictions with respect to magnetic tapes; EXECUTE and DELETE access are meaningless. Moreover, users in the SYSTEM and OWNER category are always given both READ and WRITE access, regardless of what you specify in a protection code. Protection must be explicitly specified when a volume is initialized, or all users will have READ and WRITE access. If you give WRITE access to the GROUP or WORLD categories, READ access is also allowed. For magnetic tapes mounted with the /FOREIGN qualifier, users in the SYSTEM and OWNER categories are always given logical and physical I/O access in addition to READ and WRITE access, regardless of what you specify in the protection code.

You can only change file protection on a given magnetic tape if you reinitialize the tape.

---

### 4.2.8.2 Establishing and Changing Directory Protection

UIC-based directory file protection pertains only to disk directories and is normally established when the directory is created. At directory creation time, the creator can either specify a protection code with the /PROTECTION qualifier to

## File Protection Features

the DCL command CREATE/DIRECTORY, or can permit the protection to default to that of the next higher directory in the tree. If the directory is a top-level directory, the protection is taken from the MFD, the master file directory. For example, to create the top-level directory file MONROE.DIR with open access to all but the WORLD category of users, the security manager could issue the following command:

```
$ CREATE/DIRECTORY/PROTECTION=(S:RWED,O:RWED,G:RWED,W) -  
$_BOTANYDISK: [MONROE]
```

Any user with CONTROL access can change the protection on the directory with the DCL command SET PROTECTION. For example, the following command changes the protection for the directory [MONROE] to remove access for the GROUP category:

```
$ SET PROTECTION=(S:RWED,O:RWED,G,W) MONROE.DIR
```

### 4.2.8.3

#### Establishing and Changing File Protection

When you create a new file, the file obtains a UIC-based protection code derived from the default protection provided by the directory it will reside in or the default protection of your process. (These defaults are described in further detail in Section 4.5.)

When you create a new version of an existing file, the new file receives the protection code of the previous version of the file. You can specify a protection code when you create a copy of a file. For example, you can use the /PROTECTION qualifier to define the protection for a file you create with the DCL command COPY, as shown below:

```
$ COPY USE1: [PAYDATA]PAYROLL.DAT PAYSORT.DAT -  
$_/PROTECTION=(SYSTEM:RW,OWNER:RWED,GROUP:RW,WORLD)
```

The example of the COPY command above copies a file from the device USE1 to your default disk directory. The protection code defines the protection for the newly created file PAYSORT.DAT as follows:

- Users with system UICs (those in the SYSTEM category) can read and write to the file
- You (in the OWNER category) have all types of access

## File Protection Features

- Other users in your group (those in the GROUP category) may read and write to the file
- All other users (those in the WORLD category) are permitted no access

You can also change the protection for one of your existing files with the SET PROTECTION command. For example:

```
$ SET PROTECTION=(SYSTEM:RWE,OWNER:RWED,GROUP:RE,WORLD) -  
$_PAYSORT.EXE
```

---

### 4.2.8.4 Establishing and Changing Global Section Protection

UIC-based protection on global sections, except those backed by disk files, must be re-established every time the system is booted; it is not saved. If the global section is backed by a disk file, the section protection is derived from the disk file so that changing the file protection changes the section protection. For PFN and page file global sections, you set the protection in the \$CRMPSC system service call that creates the section; you cannot change the protection after the section is created.

---

### 4.2.8.5 Establishing and Changing Device Protection

UIC-based protection on devices must be re-established every time the system is booted; it is not saved. You set device protection with the following command:

```
$ SET PROTECTION=(code)/DEVICE device-name[:]
```

---

### 4.2.8.6 Establishing and Changing Logical Name Table Protection

UIC-based protection on logical name tables must be re-established every time the system is booted; it is not saved. You set the protection with the protection argument to the \$CRELNT system service call or with the following command:

```
$ CREATE/NAME_TABLE/PROTECTION=(code) table-name
```

You cannot change the protection on an existing logical name table.

---

**4.2.8.7**

**Establishing and Changing Queue Protection**

You set UIC-based protection on a queue with the /PROTECTION qualifier to the INITIALIZE/QUEUE, START /QUEUE, or SET QUEUE command.

---

### 4.3 Access Control Lists (ACLs)

There is an alternative means of file protection offered with VAX/VMS that is known as the Access Control List. This method is used in conjunction with the standard UIC-based protection just described, as a way to fine tune that protection where it is needed. The ACL alternative offers a way to match the specific access you want to grant or deny to specific users for each object. At the heart of this method are a *rights database* that specifies *identifiers* and *holders* of those identifiers as well as ACLs that relate the identifiers with the access to be granted or denied to the holders of the identifiers. The following sections will clarify this scheme and explain how you can put it to use.

VAX/VMS provides a rights database, which is a file that associates users of the system with special names they are allowed to hold called identifiers. Some identifiers represent the users' usernames and UICs. Other identifiers are more general names that many users will hold. The security manager will maintain this rights database, adding and removing identifiers as needs change. By allowing groups of users to hold identifiers, the manager has now created a kind of group designation that differs from the one used with the user's UIC. This alternative method of grouping is more finely tailored to the uses the holders of the identifier are expected to make of the objects. This method also permits each user to be a member of multiple overlapping groups.

---

#### 4.3.1 ACLs, Identifiers, and the Reference Monitor

The reference monitor model specifies an authorization database, which describes all access authorizations in the system for all subjects and all objects. This database is often represented as an *access matrix*, listing subjects on one axis and objects on the other. Each crosspoint in the matrix thus represents the access that one subject has to one object.

Consider the example in Figure 4-2:

**Figure 4-2 Example of an Access Matrix**

Objects:	V	W	X	Y	Z
Subjects:					
A		*			*
B		*	*	*	
C		*	*	*	
D	*	*	*	*	
E	*				

In this access matrix, an asterisk (\*) is used to denote that the subject has access to that object (different types of access, such as READ and WRITE, are omitted from this example for simplicity). Thus, subjects B, C, and D all have access to objects W, X, and Y. In addition, subject A has access to objects W and Z, subject D to object V, and subject E to object V.

Few commercially available operating systems actually implement a monolithic access matrix, for reasons of performance, manageability, and control policy. The access matrix is normally distributed across the system, broken up across one of its dimensions.

For example, breaking up the access matrix by rows yields what is commonly known as a *capability based system*, in which each subject carries a list of the objects that it can access. Thus, a capability representation of this access matrix would take on the following appearance:

A: W, Z  
 B: W, X, Y  
 C: W, X, Y  
 D: V, W, X, Y  
 E: V



## File Protection Features

Alternatively, it is possible to break up the access matrix by columns, listing for each object the subjects that have access to it. This results in what is commonly known as an *authority based system*, or access control lists. The access control list representation takes on the following appearance:

V: D, E  
W: A, B, C, D  
X: B, C, D  
Y: B, C, D  
Z: A

The access control list and identifier system that VAX/VMS implements combines properties of both the capability and authority based systems. The result is an extremely powerful and flexible system that is capable of representing complex access matrices in a compact and convenient manner. Consider what happens to the previous example of an access matrix, when some of the crosspoints have labels, as in Figure 4-3.

**Figure 4-3 Previous Matrix with Labeled Crosspoints**

Objects:	V	W	X	Y	Z
Subjects:					
A	.....	..... *	.....	.....	..... *
B	.....	..... Q	..... Q	..... Q	.....
C	.....	..... Q	..... Q	..... Q	.....
D	.....	..... P	..... Q	..... Q	.....
E	.....	..... P	.....	.....	.....

Observe that some of the labeled crosspoints can be grouped and treated as a single entity. Thus, the points that are labeled Q represent the access that subjects B, C, and D have to objects W, X, and Y. All the Q points can be considered as a single area of interest. VAX/VMS provides the concept of identifiers to take practical advantage of this grouping of areas of interest, which is very common in real life situations.

## File Protection Features

With VAX/VMS, you can define identifiers to represent the two groups of access, P and Q, in the example matrix. Note that two of the crosspoints in the example remain unlabeled. VAX/VMS identifiers can also represent individual subjects, and thus allow the traditional access control list facility.

VAX/VMS uses two structures to represent the access matrix, one for each dimension. The system rights database represents the rows of the access matrix, and thus corresponds to the capability model. For this example, you would need the following rights database:

B: Q  
C: Q  
D: P, Q  
E: P

Access control lists on the protected objects represent the columns of the access matrix. For this example, you would need the following access control lists:

V: P  
W: A, Q  
X: Q  
Y: Q  
Z: A

Note that the VAX/VMS structures required to represent the access matrix are simpler than either the traditional capability or authority model, and they require fewer terms in total. In the example, the difference is slight. However, complexity of the access matrix increases with the square of its size. So, for more complicated real-life situations, the VAX/VMS system gains an advantage.

---

### 4.3.2 Creating and Maintaining ACLs

You use the VAX/VMS ACL Editor to create and edit an ACL. For information on the VAX/VMS ACL Editor, see the *VAX/VMS Utilities Reference Volume*. The DCL command SET ACL is provided to manipulate entire ACLs or individual ACEs on more than one object at a time.

The following DCL commands can be used to display ACLs:

- SHOW ACL
- DIRECTORY/ACL

- DIRECTORY/FULL

You will find that many of these commands have overlapping functionality. (In general, and in the examples throughout this guide, you will find the DCL commands SET ACL and SHOW ACL sufficient for most purposes, although the VAX/VMS ACL Editor is a important utility for more extensive ACL work.) For more information on any of these DCL commands, see the individual command descriptions in the *VAX/VMS DCL Dictionary*.

Before attempting to use any of these commands, you should understand more about the composition of ACLs and the syntax requirements for specifying them. These are the principal topics of the sections that follow.

### 4.3.2.1

#### Object Types

You can establish ACLs for various system objects: files, directory files, global sections, devices, and system logical name tables. In general, you need not be concerned about the object type when establishing or changing an ACL; however, you do need to be aware of a few object-specific considerations. This section describes those considerations.

#### Global Sections

You must re-establish ACLs on global sections (except those backed by disk files) every time the system is booted because they are not saved in any way.

The ACL on a global section backed by a file is the ACL of the file. Changing the file's ACL causes the corresponding change in the global section's ACL. The ACL on the global section itself cannot be changed directly.

You can establish ACLs on both system and group global sections. Note, however, that if a user attempts to access a group global section that is outside his UIC group, the operating system denies him access and does not consider the ACL. ACLs on PFN and page file global sections may be set up and modified with the ACL Editor or with the following commands:

```
$ SET ACL /OBJECT_TYPE=SYSTEM_GLOBAL_SECTION section_name
$ SET ACL /OBJECT_TYPE=GROUP_GLOBAL_SECTION section_name
```

### Devices

You must re-establish ACLs on devices every time the system is booted because they are not saved. ACLs on devices are set up and modified with the ACL Editor or with the following command:

```
$ SET ACL /OBJECT_TYPE=DEVICE device-name
```

### Logical Name Tables

You must re-establish ACLs on logical name tables every time the system is booted because they are not saved. ACLs can be established for system logical name tables but not process logical name tables. ACLs on system logical name tables are set up and modified with the ACL Editor or with the following command:

```
$ SET ACL /OBJECT_TYPE=LOGICAL_NAME_TABLE table-name
```

---

### 4.3.3 Types of Identifiers

Identifiers provide the means of specifying the users in an ACL. There are three types of identifiers:

- *UIC identifiers* that depend on the user identification codes (UICs) that uniquely identify each user on the system. Typically the UIC identifiers are presented in numeric format or abbreviated alphanumeric format. For example, a UIC identifier might adopt the numeric format of the UIC, such as [306,210], or just the member name from the alphanumeric format UIC, such as JONES where the full alphanumeric UIC is [GROUP1,JONES].
- *General identifiers* defined by the security manager in the system rights database to identify groups of users on the system. (Examples could be TERM3BIO, WARD5WORKERS, DATAENTRY, and RESERVDESK, which would serve as convenient ways to identify the third term biology students, the campaign workers for ward 5, the data entry personnel, or the people who handle the reservations desk, respectively.)
- *System-defined identifiers* that describe certain types of users based on their use of the system. (Examples include BATCH, NETWORK, DIALUP, INTERACTIVE, LOCAL, and REMOTE, which correspond directly to the descriptions in Section 3.1.1 of the type of login the user executed.)

---

#### 4.3.3.1 UIC Identifiers

UIC identifiers conform to the specifications for UICs, as presented in the *VAX/VMS DCL Dictionary*. Note that while the most common types of UIC identifiers are either numeric format UICs or usernames, full alphanumeric UICs or UICs in hexadecimal format are accepted as UIC identifiers. Thus, you may find UIC identifiers that assume any of the following appearances:

[PROGRAMMERS,J_JONES]	{alphanumeric format UIC}
J_JONES	{username from alphanumeric format UIC}
[341,311]	{numeric format UIC}
%X08001006	{hexadecimal format UIC}

Each of these formats serves the purpose of a UIC identifier by uniquely identifying a user.

---

#### 4.3.3.2 General Identifiers

A general identifier, defined in the system rights database, is an alphanumeric string of 1 through 31 characters that must contain at least one alphabetic character. It can include the characters A through Z, dollar signs (\$) and underscores (\_), as well as the numbers 0 through 9.

The security manager uses AUTHORIZE to create and assign the general identifiers and UICs to the system users.



---

#### 4.3.3.3 System-Defined Identifiers

System-defined identifiers are automatically defined by the system when the rights database is created at system installation time. The following identifiers, which correspond directly to the login classes that Section 3.1.1 describes, are system-defined identifiers:

BATCH	All access attempts made by batch jobs
NETWORK	All access attempts made by DECnet tasks
INTERACTIVE	All access attempts made by interactive processes
LOCAL	All access attempts made by users logged in at local terminals
DIALUP	All access attempts made by users logged in at dialup terminals
REMOTE	All access attempts made by users logged in through a network

A user automatically becomes a holder of one of these identifiers during login. That is, the VAX/VMS Login software adds the appropriate identifier to the process *rights list*.

---

#### 4.3.3.4 Storage of Process Identifiers in the Rights List

When you log in, the identifiers you hold in the rights database (including your UIC and your system-defined identifiers) are copied into a rights list that is part of your process. The rights list is the structure that VAX/VMS uses to perform all protection checks. Additional identifiers may appear in your rights list; they were put there either by the VAX/VMS Login software or by software specific to your installation. These identifiers represent various qualifications about your login, the state of the system, and so forth.

#### 4.3.4 Types of Access Control List Entries

Once the security manager has completed any necessary groundwork to customize the rights database, the next step is to specifically define which access to grant or deny to the holder of the identifier, for each object that needs this level of protection. Because there may be more than a few identifiers required to represent differing access needs for each object, it is typical to create a list of multiple entries. Each entry defines the group of access rights to be granted or denied the holders of the identifier named in that entry. Such a list is, in fact, the Access Control List, or ACL. Each entry in this list is called an access control list entry or *ACE*.

Just as it is possible to set up defaults for UIC-based protection on VAX/VMS, it is possible for security managers to set up default ACLs. As a result, some users may be unaware that their files have ACLs and may never take steps to change the ACLs themselves. Other users will be actively involved in creating and maintaining their own ACLs. To summarize, ACLs may be created by the system by default, by the security manager for specific objects, and by users to protect their own files. Unless a user owns an object or can obtain the same access as the owner, the user cannot create an ACL for that object.

An ACL consists of ACEs that grant or deny access to a particular system object, such as a file, directory, or device. Because ACLs can define access more selectively than UIC-based protection, ACLs offer users the chance to fine tune the action taken when access is requested for an object. You can provide an ACL on any file or device to permit as much or as little access as you deem desirable in each case. Typically, you use ACLs when you want to provide access to one of these objects for some, but not all system users in a way that differs from their UIC-based groups. However, ACLs can perform other functions such as dictating that security alarms be set off when access to an object succeeds or fails. This is a less common requirement and one that is enabled primarily at the security manager's direction, but it should not be overlooked in any discussion of ACLs.

Whenever the system receives a request for access to an object that has an ACL, the system searches each entry in the ACL from the first to the last for the first match it can find. It stops searching at the first match. If another match exists further



## File Protection Features

down in the ACL, it has no effect. Thus, ACEs that identify specific users should appear in the ACL before ACEs that identify broader classes of users.

The use of ACLs is optional. Although the use of ACLs can enhance the security of system objects in any installation through a more detailed definition of who is allowed what kind of access, user time must be spent in creating and maintaining the ACLs, and processor time is required to perform the functions that ACLs mandate.

Each ACL consists of one or more ACEs. There is no limit to the number of ACEs that an ACL can contain or to the number of characters in an ACE; however, very long ACLs increase the average amount of time necessary to gain access to an object.

The type of access protection that is needed determines the type of ACE used in a given situation. There are three types of ACEs involved with security:

- 1 *Identifier ACE*—Controls the type of access allowed to a particular user or group of users.
- 2 *Default protection ACE*—Defines the default protection for a directory so that the protection can be propagated to the files and subdirectories created in that directory. (This type of ACE is applicable only to directory files.)
- 3 *Security alarm ACE*—Provides an alarm message when an object is accessed in the designated way to help alert managers to possible security threats.

The exact format of an ACE depends on its type, but all ACEs are enclosed in parentheses. In general, the format of an ACE is:

(type[,options][,access\_to\_grant])

### 4.3.4.1

#### Identifier ACE

An object can have an associated ACL that explicitly specifies the access that a particular user or groups of users are allowed. In an ACL, users are specified by identifiers. You recall there are three types of identifiers:

- 1 User Identification Code (UIC)
- 2 General identifier established by the system manager in the system rights database
- 3 System-defined identifier

An identifier ACE controls the types of access allowed to specific users based on the user's identification. The format for an identifier ACE is:

```
(IDENTIFIER=identifier[,options][,access])
```

#### Specifying Identifiers in Identifier ACEs

The first field in the identifier ACE consists of the keyword IDENTIFIER followed by up to 60 identifiers. In general, you should regard the six system-defined identifiers (BATCH, NETWORK, INTERACTIVE, LOCAL, DIALUP and REMOTE) as mutually exclusive with each other and should not attempt to use them in combination with each other. However, you may combine them with other identifiers (UICs and general identifiers). When specifying multiple identifiers, separate them with plus signs (+).

The system takes the access action you specify in the ACE only for the user who holds all the identifiers specified. For example, if you wanted to grant READ access to user [301,25] running a batch job, you would specify the identifier ACE as follows:

```
(IDENTIFIER=[301,25]+BATCH,ACCESS=READ)
```

While it is not common practice for a number of users to share the same UIC, it is likely that a number of users will share the same general identifier. Users who hold the same general identifier do not need to be in the same UIC-based group. Furthermore, a single user can be associated with a number of different general identifiers as defined in the rights database. Because users can hold numerous identifiers, the creator of an ACL has considerable flexibility in selecting sets of users and defining access capabilities for them.

## File Protection Features

For example, the user identified by the UIC [301,25] is a member of the UIC-based group 301. That user may be the only member of group 301 who is also associated with the general identifier PERSONNEL. An ACE defining a particular type of access for the users associated with the general identifier PERSONNEL grants that access to user [301,25], but not to the other members of group 301.

### Specifying Options in Identifier ACEs

The options field in an identifier ACE controls whether an ACE is propagated, can be displayed, or can be deleted. That field in an identifier ACE begins with the keyword OPTIONS, and takes one or more of the keywords shown in Table 4-1.

**Table 4-1 Identifier ACE Options**

Option	Meaning
DEFAULT	Indicates that an ACE is applied to all files created within a directory. When the ACE is propagated, the DEFAULT option is removed from the ACL of the created file. This option is valid only for directory files. A default ACE does not participate in controlling access to the directory to which it belongs. It only specifies the ACL for files created in that directory.
PROTECTED	Indicates that an ACE will be preserved when an attempt is made to delete the entire ACL. A protected ACE must be specifically deleted with the VAX/VMS ACL Editor or by specifying the ACE on the command line of one of the DCL commands for ACLs.
NOPROPAGATE	Indicates that the ACE is not propagated when copying the ACL from either one version of a file to a later version of the same file or from the parent directory to a newly created file.
NONE	Indicates that no options apply to an ACE. Although you may enter OPTIONS=NONE when you create the ACE, OPTIONS=NONE is not displayed when the ACL is displayed.

Multiple options are separated by plus signs (+). If you specify any other option(s) at the same time you specify the NONE option, the other option(s) take precedence.

### Specifying Access in Identifier ACEs

The third field in an identifier ACE specifies what type of access you are allowing the user(s) identified in the first field of the ACE. That field begins with the keyword **ACCESS** followed by a string of access actions connected by plus signs (+). Table 4-2 presents the types of access allowed in an identifier ACE.

**Table 4-2 Identifier ACE Access Types**

Access Type	Meaning
READ	Accessor can read a file, read from a disk, or allocate a device
WRITE	Accessor can read or write to a file
EXECUTE	Accessor can execute an image file or look up entries in a directory without using wildcard characters
DELETE	Accessor can delete a file
CONTROL	Accessor has the same privileges as the owner of the object
NONE	Accessor has no access to the object

### Sample Identifier ACEs

The most common type of ACL is one that defines the access to a file for a group of users. In the following sample ACL, access to a file is based on the identity of a user. **PERSONNEL**, **SECURITY**, and **SECRETARIES** are general identifiers that have been assigned to appropriate sets of users by the system manager using **AUTHORIZE**. **NETWORK** is a system-defined identifier, while **[20,\*]** and **[SALES,JONES]** are examples of UIC identifiers.

```
(IDENTIFIER=SECURITY,OPTIONS=PROTECTED,
ACCESS=READ+WRITE+EXECUTE+DELETE+CONTROL)
(IDENTIFIER=PERSONNEL,ACCESS=READ+WRITE+EXECUTE+DELETE)
(IDENTIFIER=SECRETARIES,ACCESS=READ+WRITE)
(IDENTIFIER=[20,*],ACCESS=READ)
(IDENTIFIER=NETWORK,ACCESS=NONE)
(IDENTIFIER=[SALES,JONES],ACCESS=NONE)
```

## File Protection Features

In the example above, the ACE that provides the greatest amount of file access appears first in the list. In this way, if one of the users has both the SECURITY and PERSONNEL identifiers, the user obtains maximum access rights through the first match, which is the SECURITY identifier. In this example, the user with UIC [SALES,JONES] will be prohibited from any access to the file unless that user also happens to hold one of the general identifiers (which is probably an oversight on the part of the creator of the ACL). If the ACL creator wants to be absolutely certain that the user with the UIC [SALES,JONES] could not possibly gain access to the file through the ACL, the ACE that is at the bottom of the ACL should be moved to the top.

Notice also that the order of the ACEs in the example above permits a number of users to gain file access over the network. Specifically, it is the users who hold the identifiers of SECURITY, PERSONNEL, SECRETARIES, and/or UIC [20,\*] who can gain some access over the network. However, only those users who hold the identifier SECURITY can gain full access. All other users are prohibited from network access because of the fifth ACE. While this may well be the intent of the ACL creator, it would be an unfortunate oversight if it were not. Always take special care in ordering the ACEs within your ACLs.

The only ACE that includes an option field is the first ACE, which contains the PROTECTED option. Use of this option guarantees that the first ACE will not be deleted, unless its deletion is explicitly requested with the SET ACL/ACL /DELETE command or by using the VAX/VMS ACL Editor.

### Identifier ACE for a Directory

The OPTIONS=DEFAULT option of an identifier ACE allows users to define one or more default ACEs for inclusion in the ACLs for files created in a particular directory. A default ACE is supplied for all new files created in that directory; any existing files are not supplied with the default ACE.

If you want all files in the directory [MALCOLM] to have an ACE that permits READ and WRITE access to users with the PERSONNEL identifier, you could include the following ACE in the ACL for the file MALCOLM.DIR:

```
(IDENTIFIER=PERSONNEL,OPTIONS=DEFAULT,ACCESS=READ+WRITE)
```

## File Protection Features

As a result of that ACE, any file created in the [MALCOLM] directory has the following ACE:

```
(IDENTIFIER=PERSONNEL,ACCESS=READ+WRITE)
```

Notice that the DEFAULT option does not appear in the file's ACE. However, any subdirectory created in the MALCOLM directory has the DEFAULT option as part of its ACE so that the default ACE can be propagated throughout the entire directory tree.

### Identifier ACEs for Other Types of Objects

You construct identifier ACEs for objects other than files and directories just like any other identifier ACEs. However, you must specify the object type with the /OBJECT\_TYPE qualifier on the SET ACL command line.

For example, suppose your company has a special letter-quality printer (TTA8) that is used for just one purpose: printing checks. As a result, the check forms are always loaded in the printer. That device should never be used for logins, and no queues are directed to it. Only one user, MGREY, is allowed READ and WRITE access to it. The security manager accomplishes these restrictions with the next two DCL commands:

```
$ SET PROTECTION=(S,O,G,W)/DEVICE TTA8:  
$ SET ACL/OBJECT_TYPE=DEVICE TTA8: /ACL=(IDENTIFIER=MGREY,ACCESS=READ+WRITE)
```

#### 4.3.4.2

### Default Protection ACE

The default protection ACE is used to ensure that a particular type of UIC-based protection is propagated throughout a directory tree. That type of ACE allows you to specify protection for a particular directory structure that is different from the default protection applied to other directories. You can only apply default protection ACEs to directory files, and PROTECTED is the only option that you can specify.

The format for a default protection ACE is:

```
(DEFAULT_PROTECTION[,OPTIONS=PROTECTED],protection_mask)
```

You specify the *protection mask* just as you would the protection code for UIC-based protection; the user categories are SYSTEM, OWNER, GROUP, and WORLD, while the access categories are READ, WRITE, EXECUTE, and DELETE. See Section 4.2.4 for more information on protection codes.

## File Protection Features

The following ACE, included in an ACL for the directory MALCOLM, sets up default protection so that any files created in the [MALCOLM] directory will allow the SYSTEM and OWNER categories READ, WRITE, EXECUTE, and DELETE access and the GROUP and WORLD categories no access:

```
(DEFAULT_PROTECTION,S:RWED,O:RWED)
```

Note that when you add or change the default protection ACE for a directory, there is no effect on the files already created in the directory; however, all new files that are created will receive the current default protection. To update the protection of all the existing files in the directory to match the new default protection, you invoke the DCL command SET PROTECTION.

### 4.3.4.3

#### Security Alarm ACE

The security alarm ACE allows you to request that a *security alarm* (a message) be sent to the security operator's terminal if a certain type of access takes place. Whenever it is important to know that an object has been accessed in a particular way, consider adding a security alarm ACE to the object's ACL. The security alarm ACE specifies the type of access to the object that you regard as significant enough to warrant sending an alarm message to all operator terminals enabled as security operators.

The action VAX/VMS takes depends on whether the alarms have also been enabled through the DCL command SET AUDIT. If this step is not taken, the alarms will never reach the operator terminals. While users can independently create alarm ACEs in their ACLs to identify events that the system should observe and report with an alarm message, users should coordinate this activity with their system's security manager. It is the security manager who possesses the SECURITY privilege, which is required to enable the alarms feature with the SET AUDIT command. Since this feature uses system resources, the security manager may be reluctant to leave it enabled at all times. If the feature is disabled, a security alarm ACE has no effect.

The format of a security alarm ACE is:

```
(ALARM_JOURNAL=SECURITY[,options][,access])
```

This type of ACE is specified by the keywords `ALARM_JOURNAL=SECURITY`. As in the `IDENTIFIER` type of ACE, the second field in the security alarm ACE begins with the keyword `OPTIONS`, which takes one or more of the keywords described in Table 4-3.

**Table 4-3 Security Alarm ACE Options**

Option	Meaning
DEFAULT	Indicates that this ACE is to be applied to any files created within a directory. The DEFAULT option is removed from the ACE of the created file. This option is valid only for directory files.
PROTECTED	Indicates that this ACE will be preserved when an attempt is made to delete the entire ACL. A protected ACE must be explicitly deleted with the VAX/VMS ACL Editor or by specifying the ACE on the command line of one of the DCL commands for ACLs.
NOPROPAGATE	Indicates that the ACE is not propagated when copying the ACL from either one version of a file to a later version of the same file or from the parent directory to a newly created file.
NONE	Indicates that no options apply to this ACE. (Although you enter <code>OPTIONS=NONE</code> when you create the ACE, <code>OPTIONS=NONE</code> is not displayed when the ACL is displayed.)

Multiple options are separated by plus signs (+). If you specify any other option(s) in addition to `NONE`, the other option(s) take precedence.

The third field in an alarm ACE controls the type of access that causes the alarm to be sent. You can specify any of the access actions shown in Table 4-4 with the `ACCESS` keyword.



**Table 4-4 Security Alarm ACE Access Types**

Access Type	Meaning
READ	Generates an alarm if an accessor attempts to read the object
WRITE	Generates an alarm if an accessor attempts to write to the object
EXECUTE	Generates an alarm if an accessor attempts to execute the object
DELETE	Generates an alarm if an accessor attempts to delete the object
CONTROL	Generates an alarm if an accessor attempts to perform control operations on the object, such as changing the protection on the object
SUCCESS	Generates an alarm for each successful attempt by an accessor to access the object with your choice of the access types READ, WRITE, EXECUTE, DELETE, and/or CONTROL (which you include in the specification)
FAILURE	Generates an alarm for each unsuccessful attempt by an accessor to access the object with your choice of the access types READ, WRITE, EXECUTE, DELETE, and/or CONTROL (which you include in the specification)

Note that for an alarm to have any effect, you must include either SUCCESS or FAILURE (or both) in the alarm ACE. Thus, to request an alarm for successful access to write to or delete a file, you would specify the following ACE:

(ALARM\_JOURNAL=SECURITY,WRITE+DELETE+SUCCESS)

### 4.3.5 Managing Access Control Lists

There are a few common mistakes users make when first beginning to construct ACLs. By observing the following recommendations, you can avoid many pitfalls:

- Do not assume that specifying ACCESS=NONE for an identifier will absolutely prohibit the holder(s) of the identifier from accessing the object. While many times this is true, remember that users who are in either the SYSTEM or OWNER category may still be entitled to whatever access the UIC-based protection affords that category. Furthermore, if the users hold special privileges, they may be granted the access requested through the privilege.
- Watch out for errors in the order of the ACL entries. Remember to place the ACEs that deny access to **specific** users at the top of your ACLs, so that the user will not obtain access by holding another identifier. However, there is an important distinction here. Sometimes you use wildcards in the UIC-format identifiers to deny access to **large groups** of users. Because this normally represents a fall-through condition, such an ACE properly belongs at the bottom of the ACL, not at the top. Place the ACEs that grant the widest **access rights** immediately after the most restrictive ACEs. This technique ensures that users who hold multiple identifiers do not obtain restricted access rights on the first match—when another identifier they hold could grant more generous rights. Remember that a user can only receive the access rights granted through the first matching identifier.
- Do not place ACLs on everything. If you set up too many defaults, you will end up with ACLs on all your files. This is not normally necessary even at medium-level security sites. If all the users compound this error, performance penalties may appear on the system. ACLs should be applied to objects with discretion and with an understanding of how the object might be used or abused. Defaults are useful, but users should find ways to group files so that only a few directories need have default ACEs that propagate to many or all files.
- Use general identifiers to create practical groups of users to avoid unnecessarily long ACLs. Without general identifiers, users find themselves enumerating many individual users in the ACL with the same access rights.

- Update the ACLs when users leave. Users should make it a goal to maintain the shortest and most current ACLs possible. Again, using general identifiers instead of individual users helps to alleviate this problem.

---

### 4.4 Establishing and Changing Ownership

From the discussions of protection codes, it is clear that the ownership of an object is critical to the outcome of a protection check. Thus, outsiders who want to overcome a protection check may focus their energies on changing the ownership—if their attempts at changing the protection code prove unsuccessful. However, changing the ownership is usually just as challenging a task.

This section describes how VAX/VMS establishes the ownership of resources such as volumes, directories, and files. It includes an explanation of the resource attribute that users may have associated with some of their identifiers and how that attribute can affect the default file ownership. The section also describes the requirements VAX/VMS imposes on users before allowing them to change the ownership. The appropriate DCL commands to use to change ownership appear in the discussions and examples.

---

#### 4.4.1 Understanding the Role of the Resource Attribute

When your security manager tells you which identifiers you hold, you may also be told that you have the *resource attribute* with certain identifiers. This simply indicates that the security manager wants to separately track your usage of disk space associated with such identifiers. It has little noticeable effect on users, but it does introduce a special ownership relationship when directories a user accesses are owned by those identifiers for which the user also holds the resource attribute. This relationship need not be mysterious. First, consider some background information.

The consumption of disk space is monitored by VAX/VMS according to the owner UIC. Generally, this means that the creator of the file is charged for the space the file uses. (Charging refers here to subtracting the space from the user's disk quota.) However, it is not hard to imagine that system

and security managers might prefer to track the use of disk space according to logical groups of users such as departments or projects rather than individual users. You have seen how general identifiers can specify just such groups. Thus, it is advantageous to allow these types of identifiers to own files. What is needed is a mechanism to associate the fact that when a user creates a file in a directory owned by the identifier, that identifier (rather than the user's UIC) should become the owner of the file. Giving the user the resource attribute for the identifier accomplishes just that. It has the added advantage of allowing security managers to specify a subgroup of users within the group of holders of the identifier—those who are allowed to charge disk space to the identifier. That is, not everyone who holds the identifier will necessarily also hold the resource attribute with that identifier.

To summarize, to allow for more flexible management and accounting of disk space, identifiers can carry the optional resource attribute. This attribute, when present on an identifier, allows file space to be owned by and charged to that identifier. Thus, when files belong to a department or project, the space they consume can be charged to the appropriate department or project, rather than to the individual who created them. When users work on multiple projects, they can cause their disk space requirements to be charged to the related project rather than to their personal accounts. An understanding of this concept is important background for the discussions of both directory and file ownership that occur later in this section.

---

### 4.4.2 Defining the Conditions that Convey Ownership Privileges

Descriptions throughout this section refer to the conditions that qualify a user to set or change the ownership of an object. The user who meets these conditions can be said to have *ownership privileges* to the object. That is, the user may or may not be the recorded owner for the object, but by virtue of possessing a VAX/VMS privilege or holding an identifier with the resource attribute, the user may be entitled to set or change the ownership.

The general conditions that would convey ownership privileges for actions involving directories and files are identical and appear below. (The conditions for ownership privileges to volumes differ, so they appear in Section 4.4.3.)

Users would only need to satisfy one of these conditions to receive ownership privileges:

- 1 Qualify as members of the SYSTEM user category, hold SYSPRV or BYPASS privilege, or hold a UIC that matches the owner of the volume that contains the file or directory.
- 2 Hold the GRPPRV privilege while also holding a UIC in the same group as the object's owner.
- 3 Possess owner identifiers in their rights list with the resource attribute that correspond to the owner identifier. (Section 4.4.1 introduces the concept of resource attributes for identifiers.)

Note that the identifier for which ownership privileges are required depends upon the context of the proposed action. Specifically, if the user wants to establish the ownership for a new file or directory, the identifier in question is the identifier of the new file or directory. However, if the user wants to change the ownership of an existing object, the user must have ownership privileges for **both** the old and the new owner identifiers.

---

### 4.4.3 Establishing and Changing Volume Ownership

The owner of a volume is established when the volume is initialized with the DCL command INITIALIZE. Unless the qualifier /OWNER\_UIC specifies an owner, the owner defaults to the process that invoked the command.

You can display the volume owner with the DCL command SHOW DEVICES/FULL.

To change the owner of a volume, you must issue the DCL command SET VOLUME/OWNER\_UIC. Only users who can match one of the following criteria can change the ownership of the volume:

- Qualify as a member of the SYSTEM category of users
- Hold the SYSPRV privilege
- Own the volume

#### 4.4.4 Establishing and Changing Directory Ownership

Ownership of directories is normally set when the directory is created through the DCL command `CREATE/DIRECTORY`. A user with ownership privileges (see Section 4.4.2) can specify the owner with the DCL command `CREATE/DIRECTORY/OWNER_UIC`. If no owner is explicitly specified, VAX/VMS assigns a default owner depending on the format used to specify the directory, as follows:

- If the directory name is in alphanumeric or subdirectory format, the ownership defaults to the owner UIC of the existing parent directory—provided the user has ownership privileges to that UIC; otherwise, the ownership defaults to the UIC of the process issuing the command.
- If the directory name is in UIC format, the ownership defaults to the UIC in the directory name.

You can display the directory file owner with the DCL command `DIRECTORY/OWNER`. For example, to check the directory file ownership for all top-level directories, you could use the following command:

```
$ DIRECTORY/OWNER [000000]*.DIR
```

Likewise, you could view the owners of all your subdirectories with the following command:

```
$ DIRECTORY/OWNER [...]*.DIR
```

However, if there are subdirectories for which you are not the owner and you fail the protection check, you will be unable to view the owner information.

If you have ownership privileges to the directory, you can change the directory file ownership with the DCL command `SET FILE/OWNER_UIC`. For example, to change the current owner of the subdirectory `[MONROE.WEATHER]` to `[CHEM4,LEONARD]`, you could issue the following command:

```
$ SET FILE/OWNER_UIC=[CHEM4,LEONARD] [MONROE]WEATHER.DIR
```

Consider the case of the user whose rights list includes the identifier `ORCHARDIST` with the resource attribute. Suppose this user `[GROWERS,CRABTREE]` (who also holds the identifiers `PESTICIDES` and `APPLEGROUP`) wants to change the owner of the subdirectory file `TIMETABLE.DIR`.

## File Protection Features

The directory [SPRAYING.TIMETABLE] is currently owned by the identifier [ORCHARDIST]. By virtue of holding the resource attribute for the identifier ORCHARDIST, user [GROWERS,CRABTREE] could change the owner of the subdirectory file to either of those other two identifiers (APPLEGROUP or PESTICIDE). For example, user [GROWER,CRABTREE] issues the following DCL command to change the owner from [ORCHARDIST] to [APPLEGROUP]:

```
$ SET FILE/OWNER_UIC=[APPLEGROUP] [SPRAYING]TIMETABLE.DIR
```

As a result of such an ownership change, any user who holds the APPLEGROUP identifier with the resource attribute would also have ownership rights to the subdirectory [SPRAYING.TIMETABLE]. Consequently, one of these other users could also change the ownership. You can see how this might be a valuable way to extend ownership rights to a larger circle of users.

### 4.4.5 Establishing and Changing File Ownership

To find a default owner for a file, VAX/VMS considers the following sequence of choices and selects the first that yields an owner:

- 1 The owner of a previously existing version of the file
- 2 The owner of the parent directory
- 3 The UIC of the file creator

Users with ownership privileges, as described in Section 4.4.2, can specify an alternative owner with the CREATE/OWNER\_UIC command.

You can display the current file owner with the DCL command DIRECTORY/OWNER. However, if there are files in the directory for which you are not the owner and you fail the protection check, you will be unable to view the owner information.

Only those users who have ownership privileges (see Section 4.4.2) can change file ownership: If you have ownership privileges, you can change file ownership with the DCL command SET FILE/OWNER\_UIC.

---

## **4.5 Propagation of Protection Defaults**

This section extends the discussion of default UIC-based protection from Section 4.2.8 to consider ACL protection defaults and how both types of protection defaults are propagated. Since there is no default volume protection (ACL or UIC-based) with VAX/VMS, this discussion centers around protection defaults for directories and files.

---

### **4.5.1 Default Directory File Protection**

Directory file protection establishes protection for the directory itself and provides default values that can be applied to the files added to the directory, when they lack specifications of their own. Both UIC-based protection and ACL protection can be placed on directory files.

---

#### **4.5.1.1 Default UIC-Based Directory File Protection**

Directory files receive their file protection codes at creation time. The directory file protection code of a subdirectory defaults to that of its next higher level directory. The default protection code of a top-level directory comes from the volume master file directory. The creator of the directory always has the option of specifying a protection code with the DCL command CREATE DIRECTORY/PROTECTION.

You can change the directory file protection by using either of the DCL commands SET PROTECTION or SET DIRECTORY /PROTECTION.

---

#### **4.5.1.2 Default ACL Protection**

Frequently it is efficient to set up one ACL that should apply to every new subdirectory file to be created in the directory tree. By default, a subdirectory inherits the ACL of its parent. You can decide that every ACE in the ACL should be propagated, or only certain ones. You use the OPTIONS specification in the ACE to define this.

With this technique you can also provide default ACL entries for propagation into ACLs for the files kept in the directory.



---

## 4.5.2 Default File Protection

Files need UIC-based protection and sometimes require ACL protection as well. It is important to recognize how defaults for the protection code and ACLs might arise, as well as how to override them.

---

### 4.5.2.1 Default UIC-based Protection

If you do not define a protection code for a file when you create the file, the system applies a default protection code.

When you create a file, the protection is determined through the following sequence of steps:

- If the file is a new version of an existing file, the new file inherits the protection of the existing version.
- If no previous version exists and the directory where the file is to be stored has an associated ACL that includes a DEFAULT\_PROTECTION entry, the protection specified by that ACE is used.
- If the directory does not have a DEFAULT\_PROTECTION ACE, the *default process protection* is used. VAX/VMS consults the SYSGEN parameter RMS\_FILEPROT to establish this value during login. However, the value derived at login may have been overridden explicitly by you (or possibly by your login command procedure) with the DCL command SET PROTECTION/DEFAULT.

You can determine your current process default protection by issuing the DCL command SHOW PROTECTION, as follows:

```
$ SHOW PROTECTION
SYSTEM=RWED, OWNER=RWED, GROUP=RE, WORLD=NO ACCESS
```

The system responds by displaying your default protection. In this case it indicates that users in the SYSTEM and OWNER categories have all types of access, that members of the owner's group (in the GROUP category) have READ and EXECUTE access, and that all other users (in the WORLD category) have no access.

At any time during your terminal session you can change the default process protection applied to files that you create. Simply invoke the DCL command SET PROTECTION /DEFAULT.

## File Protection Features

To determine the current protection associated with a specific file or files, use the /PROTECTION qualifier with the DCL command DIRECTORY. For example:

```
$ DIRECTORY/PROTECTION PERSONNEL.REC
Directory USE1:[CRAMER]
PERSONNEL.REC;5      (RWED,RWED,RW,R)
Total of 1 file.
```

### 4.5.2.2

#### Default ACL Protection

When you create a file, it may receive an ACL by default from one of the following sources:

- If the file is a new version of an existing file, the new file inherits the ACL of the existing one, less any ACEs marked with the NOPROPAGATE option.
- If no previous version of the file exists, VAX/VMS checks the directory in which you are creating the file for an ACL. If there is an ACL, all entries in the ACL that are marked with the DEFAULT option are copied to the new file, with the DEFAULT option removed.

In addition, when you create a file whose owner identifier is not your UIC (by explicitly giving a file owner, or through the ownership defaulting rules presented in Section 4.4.5), an ACE is added to the ACL for the file that grants full access to your UIC. This feature guarantees that you retain control over a file that you have just created, regardless of other defaults.

You can change this default behavior by changing the ACLs. It is helpful to be able to use wildcard specifications in the DCL commands for ACLs to make changes on a large number of files at once. For example, if you decide you want to add an ACE to every ACL in your top-level directory that will grant holders of the identifier SPECIAL both READ and CONTROL access, you could issue the following command:

```
$ SET ACL/ACL=-
$_ (IDENTIFIER=SPECIAL,ACCESS=READ+CONTROL) *.*;*
```

Another very convenient technique when you work with ACLs is to find an existing ACL that you want to copy. You can use the DCL command SET ACL/LIKE to place a copy of an existing ACL on one of your files. For example, to place the

## File Protection Features

ACL that currently exists on the file NEWTERM.MEM onto the file FALLTERM.MEM, you could issue the following command:

```
$ SET ACL/LIKE=NEWTERM.MEM FALLTERM.MEM
```

Even if the ACL requires a small amount of editing, you may save considerable typing if you copy it first. Since the SET /ACL/LIKE command will also accept wildcard specifications, you could rapidly propagate the final, edited version of the ACL to many files.

---

### 4.5.2.3 File Protection Considerations

It is not sufficient to protect a file. You must ensure the protection for its directory is consistent with your intent.

The following example shows how a user who has READ and WRITE access to the directory where a file resides can actually remove the file from its directory even when the file protection denies that user DELETE access:

```
$ RENAME [JONES]STATUS.RNO [SMITH]JUNK.XYZ
```

The file STATUS.RNO has been renamed JUNK.XYZ, in the directory [SMITH]. Although the file still exists, it is no longer cataloged in its owner's directory [JONES]. Unless a directory and a file are both protected, processes that have WRITE access to a directory can use the DCL command RENAME to "delete" any file from that directory—even if the file is protected against deletion.

**Note:** To protect a file totally, you must protect both the file itself and the directory in which the file is listed.

---

## 4.6 Summary of File Protection Evaluation

All the components that influence whether or not a requester gains access to an object have been described in detail. It is now possible to summarize their relationships to one another. Figure 4-4 flowcharts the sequence of procedures that VAX/VMS follows when evaluating an access request. From this diagram you can learn the way the three controlling components (ACLs, protection codes, and privileges) interact. You might try working out a few cases of file access requests to test your understanding of the outcome of a VAX/VMS protection check.

## File Protection Features

For example, assume the file QUALITY\_ASSURANCE.MEM has the following characteristics:

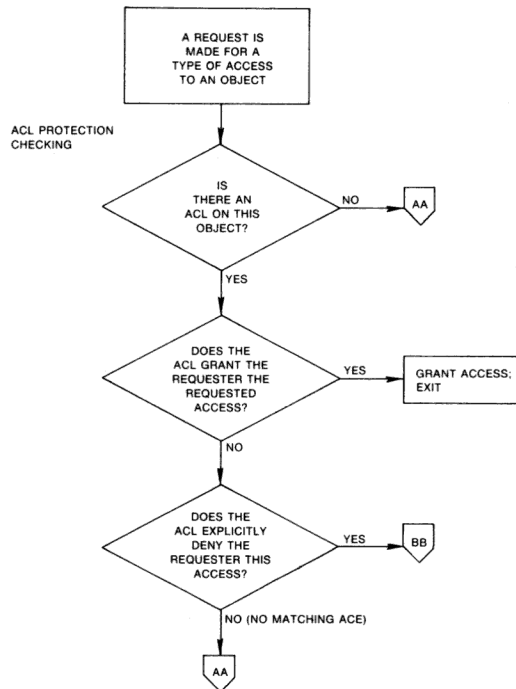
- An ACL with the following ACEs:  
(IDENTIFIER=QAMANAGEMENT, ACCESS=READ+WRITE+EXECUTE+DELETE)  
(IDENTIFIER=MADISON, ACCESS=READ+CONTROL)  
(IDENTIFIER=GRANGER, ACCESS=READ)  
(IDENTIFIER=[\*,\*], ACCESS=NONE)
- An owner UIC of [QAMGT,MORRIS]
- A protection code of (S:RWED,O:RWE,G:R,W)

The following users want to access QUALITY\_ASSURANCE.MEM in particular ways. Will they succeed or fail? (Assume that if no mention is made of a privilege or particular user category, the user does not have it.)

- 1 MADISON wants to print then delete it
- 2 MORRIS who has SYSPRV wants to delete it
- 3 GRANGER who has the UIC [QAMGT,GRANGER] and GRPPRV wants to append an addendum to it
- 4 MONROE with UIC [QAMGT,MONROE] wants to delete it
- 5 WALLACE who holds the QAMANAGEMENT identifier wants to delete it

(You can confirm your results by checking the answers provided at the end of this chapter.)

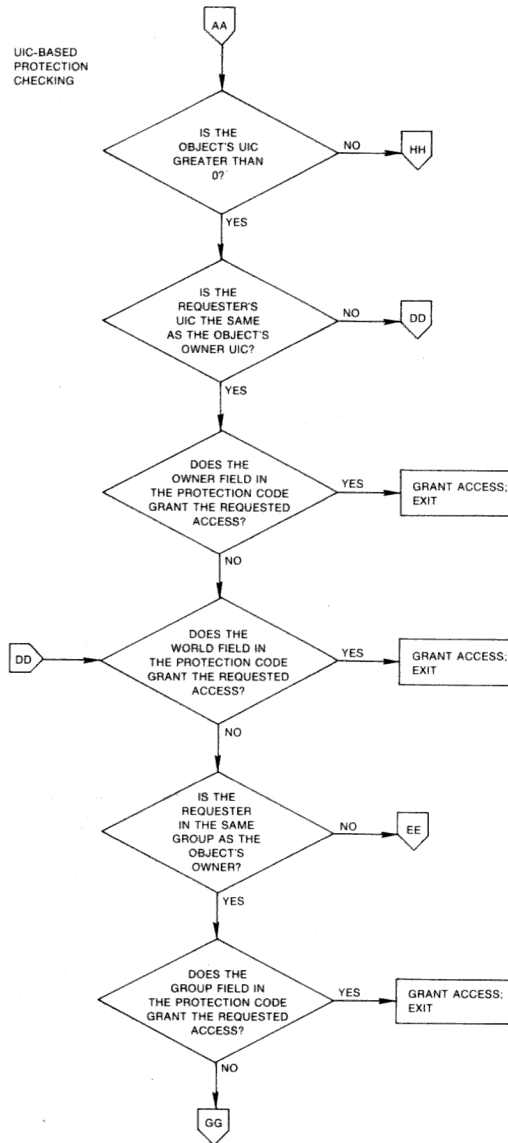
**Figure 4-4 Flowchart of Access Request Evaluation**



ZK-2039/1-84

(Continued on next page)

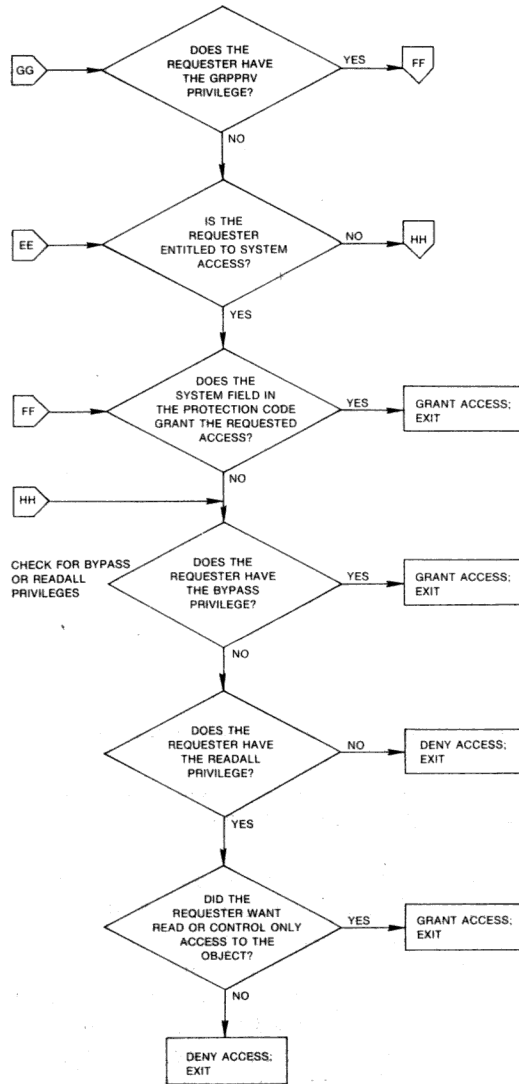
**Figure 4-4 (Cont.) Flowchart of Access Request Evaluation**



ZK-2039/2-84

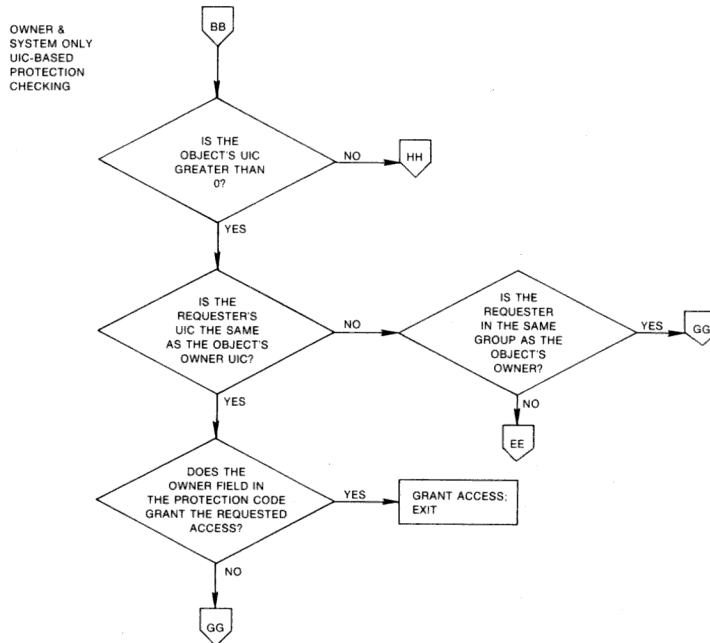
(Continued on next page)

**Figure 4-4 (Cont.) Flowchart of Access Request Evaluation**



ZK-2039/3-84

(Continued on next page)

**Figure 4-4 (Cont.) Flowchart of Access Request Evaluation**

ZK-2039/4-84

## 4.7 Escaping the Disk Scavenger

*Disk scavenging* refers to a technique used to glean information from disks. The information is old files or fragments of files that users have deleted or purged. This may sound like a contradiction, but when you delete or purge a file, the file remains on the disk until it is overwritten. However, the file's header record is changed so that the file can no longer be accessed by normal means. The scavenger, however, does not apply normal means. The scavenger uses special programs and/or equipment to read the files. In fact, the technology is so advanced that some devices can even read the faint residual magnetic impressions of files that have been overwritten. As a result, strong countermeasures may be called for at sites with medium- to high-level security requirements.





ZK-2066-84

Your security manager may take several steps to thwart disk scavengers, above and beyond trying to control those who might have physical access to the disks. One possibility involves the use of *erasure patterns*. At some sites the erasure pattern is automatically applied whenever files are deleted or purged on the disk through controls applied to the volume. At other sites, users may be asked to remember to use the `/ERASE` qualifier when invoking the DCL commands `SET FILE`, `DELETE` and/or `PURGE` for selected files.

You may also hear the term *highwater marking* used at your site in connection with the disk scavenging problem. This concept evolves from the desire to keep users from reading file space beyond the areas where they have been permitted to write. You can think of the outer limits of written space on

the file as that file's highwater mark. This technique prevents users from scavenging portions of the disk for information that they did not actually write themselves. Of course, this assumes that no one invades another user's account, bypasses controls, and so forth. VAX/VMS implements a variation of highwater marking by using an *erase-on-allocate* strategy. Since VAX/VMS enables highwater marking as the normal default at volume initialization, the security manager might decide to curtail highwater marking on a volume-by-volume basis with the DCL command SET VOLUME/NOHIGHWATER.

Highwater marking differs from erasure patterns as a method for preventing disk scavenging in several ways. You should be aware that highwater marking takes effect for entire volumes at a time. Erasure patterns can be applied to the entire volume or enabled on a file-by-file basis.

---

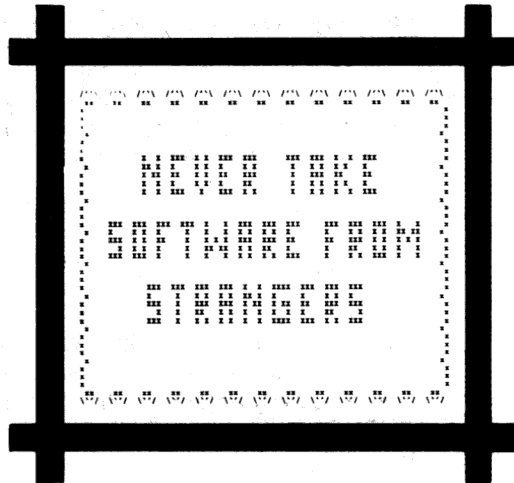
### 4.8 Managing Your Files for Optimum Security

Proper file management is an important aspect of file protection. While the following suggestions adhere to common sense, they may not be obvious to all users, and they warrant some mention as part of the subject of file protection.

- Avoid the temptation to use such obvious names for your directories and the key files in them that you will always provoke interest if a browser gains access and scans your directory. For example, do not title the file that contains a salary planning memo, SALARYPLAN.MEM. On the other hand, if all your files have obvious names except for a few, the few will draw the most attention. Use common sense here.
- Purge your files regularly and delete unnecessary files. This way you will keep your directories to a minimum and will simplify the important task of regularly checking the protection and ownership on your files, as described next.
- Use the DCL command DIRECTORY/SECURITY to monitor the ownership, protection code, and ACLs on your files. A user who succeeds in obtaining sufficient privilege may change the protection or ownership on your files allowing access both immediately and later on. If you perform these checks frequently, you will be able to readily detect and report unexplained changes in the file protection or ownership.

## File Protection Features

- Pay special attention to the protection on your mail files; normally they should only be accessible to you and the system (for mail delivery and backups).
- Occasionally look at the dates your files were last revised and check the dates against your recollection of activity. Report any unexplained activity to your security manager.
- When you place ACLs on your files, be sure you know exactly which users hold the identifiers you have specified. (This will generally require some consultation with your security manager.)
- Follow your security manager's recommendations to prevent disk scavenging. You may be requested to use the /ERASE qualifier on the SET FILE, DELETE, and/or PURGE commands that you issue for some or all of your files.
- Devote special effort to protecting files and directories that contain command procedures and executable programs. A user who successfully modifies one of your files or command procedures could cause you to execute some other commands or a program—with all your rights and privileges. Thus, you should carefully control the granting of WRITE access to these directories and files. This caution is particularly important if you possess any of the more powerful privileges or have access to sensitive files.
- Similarly, do not run a command procedure or program given to you by another user unless you inspect it or have reason to trust the user explicitly. When you inspect a program or procedure, look for evidence that it tries to exercise some of your special privileges or access some of your sensitive files. Programs or command procedures offered under one guise, when actually intended to penetrate your defenses and disrupt your system security, are sometimes called *Trojan Horse programs*, because they parallel the theme of that Greek legend.



ZK-2070-84

### 4.9 Auditing for the User

The alert user can assist the security manager in *auditing* security in several ways. After all, most break-ins occur at the account level. The owner of the account is in the best position to detect and report either the actual break-in or some unusual incident. This section introduces several ways in which you can become more aware of breakin attempts and possibly assist in identifying the culprits.

#### 4.9.1 Noting Your Last Login Time

Section 3.1.2.4 describes how VAX/VMS maintains some information in your UAF record regarding the last logins into your account. Your security manager will decide for each user whether or not the system should display this information at login time. Normally sites with medium to high security requirements will choose to display this information and enlist the aid of users to check it for any unusual or unexplained successful logins, as well as any unexplained failed logins.

If your security manager chooses to display this information when you log in, you should take these messages seriously. Each time you see them, match them to your recollection of the last time you personally logged in interactively or believe your account would have been validly accessed noninteractively by a batch job.

If you observe either an interactive or noninteractive type of login at a time that seems unlikely, report it promptly to your security manager and change your password immediately. The security manager can investigate further using the system accounting and audit logs.

Equally important is the message that there were login failures for your account since the last successful login. If you can not personally account for the failures, it is likely that someone has been trying to access your account unsuccessfully. Reconsider whether your password adheres to all the recommendations for password security that Section 3.1.3.10 describes. If not, change your password immediately.

On the other hand, if you expect to see a login failure message and either it does not appear or the count of failures is too low, your suspicions should be aroused. You should immediately suspect that you have become a victim of a password grabber. Change your password as soon as possible. You should also report either of these indications of login failure problems, so that you can take any additional precautions that your security manager recommends to protect your account.

The purpose of the last login messages is to give users a chance to detect break-ins or attempted break-ins into their accounts. If users become insensitive to these messages, the messages will fail in their purpose.

---

## 4.9.2 Tools for Detecting System Abuse

The security manager has a large collection of tools to assist in detecting system abuses. Among these tools are security alarms, the VAX/VMS Accounting Utility, and the VAX/VMS Monitor Utility. As a user, you should be at least minimally aware of these tools so that you can request their use when appropriate, and can assist in their application and interpretation. (For some users it is possible the following discussion will lead to more respect for the system and less temptation to abuse it.)

---

### 4.9.2.1 Security Alarms

Some security managers may choose to implement the security alarm feature when needed. The security manager can select one or more types of events that seem to warrant special attention whenever they occur. The security manager then directs the system to send an alarm to the terminals enabled as security operator terminals whenever the event is detected. For example, the security manager might identify one or more files of such importance that all attempts to write to them should be prohibited. If anyone succeeds in acquiring WRITE access to these files, the files are alarmed so that the security manager learns the penetration has occurred or has been attempted.

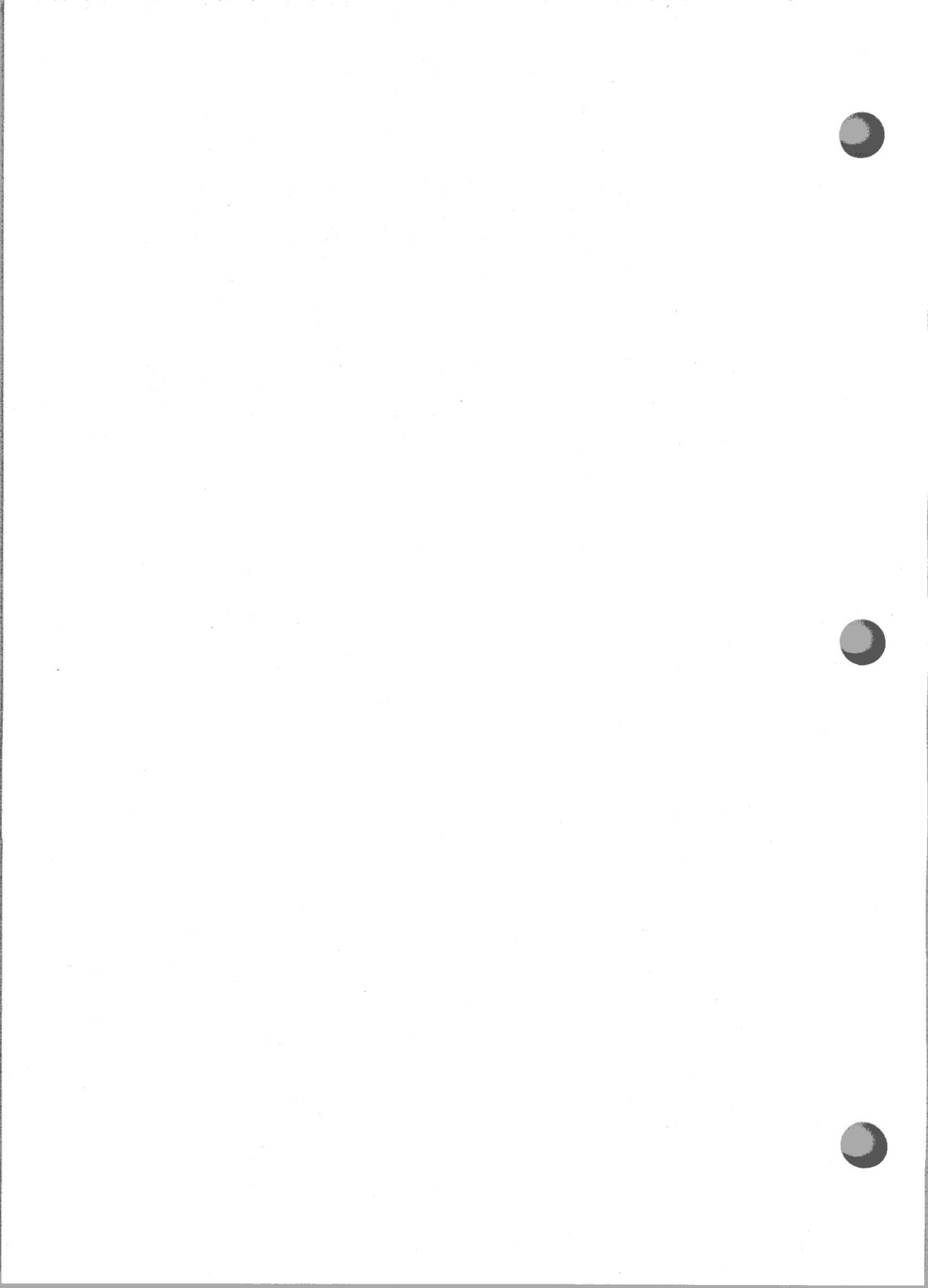
Among the events that may be alarmed are:

- Selected types of access to selected files and global sections
- Event requested by an ACL on a file or global section
- Use of privilege to access files and global sections
- Installation of images
- Logins, logouts, and breakin attempts
- Modifications to the system and network UAF
- Changes to system and user passwords
- Modifications to the rights database
- Execution of the SET AUDIT command
- Volume mounts and dismounts

## **File Protection Features**

Note that alarms can be added or removed as necessary. The best practice is to implement alarms only for key events, sometimes just a few at a time. Enabling too many alarms results in the failure to monitor each alarm appropriately and fosters lax attitudes about alarms. While alarms can be a powerful tool when used judiciously, they quickly lose their attention-getting quality when overused.

If you suspect your account has been broken into, change your password. Then consider whether you might also want to request that your security manager implement an alarm. Once an alarm has been introduced, check back from time-to-time with your security manager to see if any additional break-ins have occurred.





#### 4.9.2.2 Auditing Access to Sensitive Files

If you feel you have key files that may have been improperly accessed, discuss the matter with your security manager. Together you may want to develop a strategy to audit the access to the file.

Once you have reviewed the situation and ensured that you have done everything possible to protect your files with standard UIC-based protection and possibly a general access control list, you may conclude that a security alarm is required. To specify a security alarm, you must include a security alarm ACE in the ACL for the file.

In a case such as this one where your suspicions are already aroused, it might be advisable (on a temporary basis), to enable an alarm for all file accesses. (In cases where the security manager is on the alert, but not overly worried, the alarm might be set only for READ access and failures, to catch browsers.)

For example, if user RWOODS and his security manager concur that they must know when RWOODS' highly confidential file CONFIDREVIEW.MEM is being accessed, RWOODS would add an ACE to the existing ACL for the file CONFIDREVIEW.MEM, as follows:

```
$ SET ACL CONFIDREVIEW.MEM /ACL=-  
$_(ALARM=SECURITY,ACCESS=READ+WRITE+DELETE+CONTROL+FAIL+SUCCESS)
```

The security manager might then approach a terminal suited to security alarms and issue the following DCL commands:

```
$ REPLY/ENABLE=SECURITY  
$ SET AUDIT/ENABLE=ACL/ALARM
```

The first command causes the terminal that issues the command to be enabled as a security operator. For the sake of example, assume no terminals have been established as security operator terminals and this terminal is TTA9. The second command enables the auditing of security alarms for file accesses involving access control list alarm ACEs. The second command could be issued by the security manager from any terminal. After a while, the user ABADGUY accesses CONFIDREVIEW.MEM with DELETE access, and the following alarm appears at the security operator terminal TTA9:

## File Protection Features

```

%%%%%%%%%% OPCOM 16-AUG-1985 07:21:11:10 %%%%%%%%%%
Security alarm      / Successful file access
Time:              16-AUG-1985 07:21:10.84
PID:               23E00231
User Name:         ABADGUY
Image:             DUA0: [SYSO.] [SYSEXE]DELETE.EXE
File:              DUA1: [RWOODS]CONFIDREVIEW.MEM
Mode:              DELETE
Privs Used:        SYSPRV

```

You can see that the alarm reveals the name of the perpetrator, the method of access (successful deletion accomplished by using the program [SYSEXE]DELETE.EXE), time of access (early in the morning at 7:21 a.m.), and the use of a privilege (SYSPRV) to gain access to the file. Armed with this information, the security manager is now in a better position to take corrective action.

Note that the security alarm appears at **each** terminal enabled as a security operator, which in this case is just TTA9, **every** time **any** file is accessed through the use of the conditions specified in the alarm ACE for that file. Since the file CONFIDREVIEW.MEM is not the only file with an alarm ACE, this approach is not very specific to CONFIDREVIEW.MEM. However, access to CONFIDREVIEW.MEM does evoke the alarm illustrated above, along with some others.

It may prove helpful for the security manager to observe the access to all the key files armed with alarm ACEs. It is not uncommon to find access problems on multiple files when there are problems with one file. Whenever it becomes obvious that a significant amount of undesired access is being gained to key files, the security manager must take action. Any of the following possibilities could be at fault:

- The protection schemes may be poorly designed.
- Too many users may have too many privileges or the wrong privileges.
- Users might be abusing their privileges.
- Unwanted users might be usurping privileges.
- Other aspects may be out of balance.

The security manager would want to investigate these possibilities even further.

---

## 4.10 Summary of File Protection Techniques

This chapter describes file protection techniques that include setting a protection code to restrict the access to an object, placing an access control list on the object to define which users will gain specific types of access, and setting alarms to detect when the object is accessed improperly. The topics of defaults for protection and ownership are discussed. You may also find it helpful to adjust the default protection placed on your files—to make the protection of your files both more automatic and more appropriate to your intended uses.

---

## 4.11 Answers to the Protection Checking Self-test

The following solutions and explanations pertain to the self-test in Section 4.6.

- 1 MADISON fails. The ACL will only allow MADISON READ access to the file and PRINT/DELETE requires both READ and DELETE access. You must assume that MADISON has no privileges and belongs to no special user category.
- 2 MORRIS succeeds. The UIC-based protection allows DELETE access to the SYSTEM category, which MORRIS is entitled to through the SYSPRV privilege.
- 3 GRANGER succeeds. The ACL denies WRITE access, but GRPPRV entitles GRANGER (who is in the same group as the owner) to READ and WRITE the file through the SYSTEM category.
- 4 MONROE fails. MONROE matches the wildcard ACE that denies access. Furthermore, MONROE is not in the SYSTEM or OWNER category and lacks privileges, which means there is no way to circumvent the restrictions from the ACE.
- 5 WALLACE succeeds. WALLACE holds the identifier QAMANAGEMENT that grants DELETE access.



# 5

## Security for the System Manager

---

While Chapters 3 and 4 introduce a number of interesting security concepts and user techniques, they reveal little of how a security manager would implement the features. In this chapter, security managers will learn how to put the features into practice. The primary tool in this task is the VAX/VMS Authorize Utility (AUTHORIZE), which is described in both the *Guide to VAX/VMS System Management and Daily Operations* and the *VAX/VMS Utilities Reference Volume*. Other tools include: the VAX/VMS System Generation Utility (SYSGEN), which is described in both the *Guide to VAX/VMS System Management and Daily Operations* and the *VAX/VMS Utilities Reference Volume*, as well as numerous DCL commands that are included in the *VAX/VMS DCL Dictionary*.

While Chapter 1 demonstrates the diversity of security levels required at various sites, this chapter assumes the model of a security manager at an average commercial installation that has files and accounts that require protection. Throughout this chapter, when discussions lead into security techniques that exceed the average level of requirement, the distinction will be noted.

You will face some of the choices that this chapter describes on a recurring basis as you add new users. Other choices affect the entire installation and need only be resolved once. As you read this chapter, do not stop to implement any specific ideas. Read the entire chapter and plan to spend some time developing an overall philosophy to suit your site. Return to the chapter and review your developing philosophy against each topic. Use the chapter as a checklist to see how well you are incorporating the ideas into your own framework. Once you have taken these steps, and possibly discussed your ideas with others at your site, start to implement the ideas. You will save time in the long run, and your policies will be both more consistent and more enduring.

---

## 5.1 Security Management Account

Security managers require accounts with privileges. In many cases the security manager role and system manager role will be performed by the same individual, so the same account can serve both purposes. The *Guide to VAX/VMS System Management and Daily Operations* describes the necessary characteristics of a system management account. The same principles apply to the security management role. It is important that strong cooperation and open communication exist when the security management and system management roles are performed by separate individuals. One way to facilitate this is to have separate accounts so that electronic mail can be sent, but also provide the individuals with access to each other's files. The security manager requires the additional SECURITY privilege.

---

## 5.2 Considerations for Establishing User Accounts

As a security manager, one of the activities you will engage in most frequently is interfacing with the users, training them, adding accounts for them, assigning their initial passwords, and modifying their accounts to add or remove restrictions. This section will discuss your major considerations in these activities and provide some recommended guidelines.

The primary tool for all these activities is AUTHORIZE, which is described in both the *VAX/VMS Utilities Reference Volume* and the *Guide to VAX/VMS System Management and Daily Operations*. Section 5.3.7 illustrates three kinds of user accounts created with AUTHORIZE. The examples range from highly privileged, to moderately privileged, to extremely restricted. The examples are not intended as templates; they are provided for discussion.

Each security manager must apply individual insight into the site's operation when deciding how to set up user accounts. It is likely you can develop one or more of your own templates that work for many of your users. However, you should refrain from oversimplifying the process of account creation to the point that you simply apply a template. The danger in relying solely on templates is overlooking special considerations that apply to individual users, so that you forfeit important controls that only you can exercise.

If you do use templates, plan to reexamine them every so often to be sure they are valid and reflect the way you want your operations to proceed. Templates, in addition to oversimplifying the problem, do become obsolete much more rapidly than you might guess until you gain experience working with them.

---

### 5.2.1 Introduction to Group Design

As you contemplate the placement of users in groups, remember that the groups you compose will have impact on file protection and will influence those who receive the GROUP, GRPNAM, and GRPPRV privileges. Sometimes it is helpful to first map out the functions you expect your users to perform. Look for common groups of users involved with a function.

As you perform this exercise, think ahead, too, to what you may know about future plans in your organization. Try to incorporate these ideas into your strategy. This is not an easy job, but it is worth the effort you put into it. You can fine tune the groups at any time, but the most important step you will take is to gain perspective on the logical groupings according to the functions your users perform.

The following example will illustrate many of the principles of group design. Assume the Rainbows-4-U Paint Company is a distribution company with five different departments: executive, accounting, marketing, shipping, and secretarial. The informal table in Figure 5-1 identifies the employees in those departments who need computer resources and their job responsibilities.

**Figure 5-1 Employee Grouping by Department and Function**

Department	Employee	Function
EXECUTIVE	Sunny Gold	President
	Olive Green	Treasurer, Head of Computer Operations
ACCOUNTING	Lily White	Payroll
	Rich Silver	Bookkeeping
	Violet Indigo	Clerk
	Ruby Crimson	Clerk
MARKETING	Rusty Brown	Forecasting
	Tawny Copper	Sales Reporting
SHIPPING	Cole Black	Inventory Control
SECRETARIAL	Misty Grey	Correspondence Management
		Paycheck Printing

The fact that the company has been organized into departments suggests that individuals in the same department perform many of the same functions. For example, the advantage of grouping all the employees who "keep the books" for the company in the accounting department is that employees can easily gain access to one another and to the data that they must share.

As the system manager of Rainbow Paint's computer resources, Olive Green will set up UIC groups based on the existing organizational structure. For example, the employees in the accounting department (L. White, R. Silver, V. Indigo, and R. Crimson) could be members of the UIC group ACCOUNTING. Setting up the UIC group in this way ensures user L. White has easy access to data from user R. Silver, and so forth.

Another reason that Rainbow Paint is organized into departments is that everyone in the company should not have access to all the data and employees in the company. For example, one of the functions of the accounting department concerns the payroll. Because payroll information is confidential information, the employees in the shipping and marketing departments should not have access to that information.



As the system manager, Olive must also consider how to protect confidential information. By setting up the UIC groups according to the departmental organization, Olive can protect the data according to those groups. For example, the UIC groups MARKETING and SHIPPING can be denied access to data and users in other groups by the way that access for different groups is specified.

When determining the placement of users in UIC groups, use these two guidelines:

- **Sharing**—Users who typically share data and/or control of one another's processes should be arranged in the same group.
- **Protection**—Users who should not have access to each other's data or control each other's processes should be assigned to separate groups.

As the system/security manager of Rainbow Paint's computer resources, Olive decides to set up the UIC groups ACCOUNTING, EXECUTIVE, MARKETING, SHIPPING, and SECRETARIAL. However, when she considers how payroll is done at the company, she becomes aware of certain problems. As the company treasurer, Olive needs access to accounting data. The president of the company, Sunny Gold, also wants access to that information. If Olive allows world access to the accounting data so that she and Sunny can access it, all the users on the system will also be able to access the data, effectively destroying the protection. Thus, in this particular case, UIC-based protection does not present the best possible solution.

---

### 5.2.2 Introduction to ACL Design

While considering this problem further, Olive realizes that Misty Grey in the SECRETARIAL group needs access to payroll information so that she can type out the checks. A further complication is that the clerks in the accounting department, Violet Indigo and Ruby Crimson, should not have access to the more confidential pieces of payroll data. Since they are part of the UIC group ACCOUNTING, however, they would be granted access through UIC-based protection. Once again, the evidence suggests that UIC-based protection does not present the best possible solution.

Figure 5-2 defines the access requirements for one object involved in the payroll process, the file PAYROLL.DAT, including the users who need to access it and their access requirements.

**Figure 5-2 Access Requirements for Payroll Application**

User	UIC Group	Access Needed
SGOLD	EXECUTIVE	R
OGREEN	EXECUTIVE	RWED
LWHITE	ACCOUNTING	RWED
RSILVER	ACCOUNTING	RWED
VINDIGO	ACCOUNTING	None
RCRIMSON	ACCOUNTING	None
RBROWN	MARKETING	None
TCOPPER	MARKETING	None
CBLACK	SHIPPING	None
MGREY	SECRETARIAL	R

When you extend this problem to a few other objects, such as the order processing database and the accounts receivable program, you quickly realize it is impossible to structure groups to afford the protection you want. For example, the order processing database might require the access depicted by Figure 5-3.

**Figure 5-3 Access Requirements for Order Processing**

User	UIC Group	Access Needed
TCOPPER	MARKETING	RWED
OGREEN	EXECUTIVE	RWED
CBLACK	SHIPPING	RW
RBROWN	MARKETING	R

As shown in Figure 5-4, the accounts receivable program might require yet a different type of access from the different groups.

**Figure 5-4 Access Requirements for Accounts Receivable**

User	UIC Group	Access Needed
OGREEN	EXECUTIVE	RWED
RSILVER	ACCOUNTING	RWED
VINDIGO	ACCOUNTING	R
TCOPPER	MARKETING	R
CBLACK	SHIPPING	RW

### 5.2.3 Introduction to Identifier Design

Rather than attempt to restructure the UIC groups around this situation, Olive decides to achieve her goals by using access control lists on the files. For example, and to simplify the situation somewhat, consider the ACL that Olive might construct for just PAYROLL.DAT:

```
(IDENTIFIER=OGREEN, ACCESS=READ+WRITE+EXECUTE+DELETE)
(IDENTIFIER=LWHITE, ACCESS=READ+WRITE+EXECUTE+DELETE)
(IDENTIFIER=RSILVER, ACCESS=READ+WRITE+EXECUTE+DELETE)
(IDENTIFIER=MGREY, ACCESS=READ)
(IDENTIFIER=SGOLD, ACCESS=READ)
```

Notice that many of the users share the same access needs. There is an alternative that will shorten the ACL. Olive could use AUTHORIZE to define a general identifier PAYROLL in the rights database. The holders of that identifier could be all the users who need RWED access to PAYROLL.DAT. Once the identifier and its holders are defined, Olive could then use the following simpler ACL to specify the same type of access she previously allowed for PAYROLL.DAT:

```
(IDENTIFIER=PAYROLL, ACCESS=READ+WRITE+EXECUTE+DELETE)
(IDENTIFIER=MGREY, ACCESS=READ)
(IDENTIFIER=SGOLD, ACCESS=READ)
```

VAX/VMS can process shorter ACLs more rapidly. Another advantage of this approach surfaces when employees change but the functions remain the same. Olive does not have to change every ACL across the system. She removes the user's UAF record; as a consequence, that user also no longer holds the identifier. When she replaces the employee, she grants that new user the right to hold the identifier.

Since this example is simplified, the benefits of this solution may not seem very dramatic. However, consider the typical situation where there are many more users and more objects to be protected. Then, the flexibility and the advantages of using identifiers becomes more apparent.

What you want most of all is an overall scheme that considers the types of files on your system and the protection needs of each. If you have done a good job of designating groups and identifiers, you should be able to design ACLs and define standard protection with minimal pain. Time spent clarifying the common access needs of your users pays dividends in simplifying the design of identifiers and ACLs. You will also provide a framework that your users will appreciate for their own designs of ACLs on the files they control.

There are some disadvantages to ACLs. They can consume amounts of paged system dynamic memory when files are open. They also require additional processing time. Thus, you should not use them indiscriminately. They are best applied where protection is really needed. There is no other way of affording such refined protection when the situation demands it.

For more information on defining identifiers, see the description of the VAX/VMS Authorize Utility in the *VAX/VMS Utilities Reference Volume*. For more information on creating and maintaining an ACL, see the VAX/VMS ACL Editor description in the *VAX/VMS Utilities Reference Volume*.

---

#### 5.2.4 Some Special-Purpose Identifiers

The system provides a group of reserved identifiers for your convenience: LOCAL, DIALUP, REMOTE, INTERACTIVE, NETWORK, and BATCH. These identifiers permit you to define a large potential group of users according to their use of the system. Typically, these sorts of identifiers are used in combination with other identifiers. For example, the following ACE permits Olive Green to have RWED access to the object only when she is logged in from a local terminal:

```
(IDENTIFIER=OGREEN+LOCAL, ACCESS=READ+WRITE+EXECUTE+DELETE)
```

In a similar way, it is possible to deny access to all dialup users, as in the following ACE:

```
(IDENTIFIER=DIALUP, ACCESS=NONE)
```

---

#### 5.2.5 Creating and Maintaining a Rights Database

Once you have designed the names of the identifiers you want on your system and composed the set of holders for the identifiers, you must use the VAX/VMS Authorize Utility to make these associations known to the system. These associations are really the rights database, which you maintain as you add or remove users and add or remove identifiers. The rights database is initially created for every VAX/VMS system at system installation time. At that time it is in an elementary state with certain reserved system identifiers, and one identifier for each authorized user, which associates the username with the user's UIC. Furthermore, for each group there is a rights database entry where the account is equivalent to the UIC group. In fact, this explains how the display and acceptance of alphanumeric UICs is managed. Each site then further evolves its own rights database according to actual use and needs.

Note that when you use AUTHORIZE to simply add, remove, or change usernames in the system UAF, AUTHORIZE makes corresponding changes for you in the rights database, so that the database corresponds to the system UAF.

You should rarely find that you need to use the AUTHORIZE command CREATE/RIGHTS, because of the automatic creation and maintenance of the rights database. However, if the rights database is damaged or deleted, you can create a new one with this command.

#### 5.2.5.1

##### Adding Identifiers

You can add identifiers with the AUTHORIZE command ADD /IDENTIFIER. For example, using the example at Rainbow Paint, when Olive is ready to add the identifier PAYROLL, she invokes AUTHORIZE and specifies the following command:

```
UAF> ADD/IDENTIFIER PAYROLL
identifier PAYROLL value %X80080011 added to RIGHTS.LIST.DAT
```

If Olive accidentally deletes her rights database at some point in time, and is unable to recover from a backup copy, she can begin recreating RIGHTS.LIST.DAT by first issuing a CREATE /RIGHTS command and then an ADD/IDENTIFIER command as follows:

```
UAF> CREATE/RIGHTS
{message}
UAF> ADD/IDENTIFIER/USER=*
{messages}
```

The ADD/IDENTIFIER command generates a rights database at Rainbow Paint where every user's username is a valid identifier. Olive would have to complete the task by using the ADD /IDENTIFIER command to add all the other general identifiers that were lost. Then she would fully recover by associating the holders with the identifiers, with GRANT/IDENTIFIER commands, as shown in Section 5.2.5.2.

#### 5.2.5.2

##### Adding Holders of Identifiers

You can associate users as the holders of existing identifiers with the AUTHORIZE command GRANT/IDENTIFIER. Returning to the Rainbow Paint example, Olive would invoke AUTHORIZE and then issue the following commands to make Rich, Lily and herself the holders of the PAYROLL identifier.

```
UAF> GRANT/IDENTIFIER PAYROLL RSILVER
identifier PAYROLL granted to RSILVER
UAF> GRANT/IDENTIFIER PAYROLL LWHITE
identifier PAYROLL granted to LWHITE
UAF> GRANT/IDENTIFIER PAYROLL OGREEN
identifier PAYROLL granted to OGREEN
```

To also give Lily the ACCTSRCV identifier would require another use of the GRANT/IDENTIFIER command, as follows:

```
UAF> GRANT/IDENTIFIER ACCTSRCV LWHITE
identifier ACCTSRCV granted to LWHITE
```

In other words, you can only introduce one holder association at a time with the GRANT/IDENTIFIER command.

### 5.2.5.3 Removing Identifiers and Holders

When users at your site leave the company, you remove their UAF records. You should also notify the managers of all sites where they have access to proxy accounts, so that the proxy access record in the NETUAF can be removed. When you run AUTHORIZE to remove a user's UAF record, AUTHORIZE also removes the user's connections as a holder of identifiers in the rights database. However, if a departed user is the only remaining holder of a given identifier, then you generally should also take steps to remove that identifier, to avoid any future confusion.

For example, to remove the identifier 85TERM3, you would use the following AUTHORIZE command:

```
UAF> REMOVE/IDENTIFIER 85TERM3  
record removed from RIGHTSLIST.DAT
```

When you remove an identifier from the rights database, you should review the ACLs where you know or suspect that identifier was used. (This is not necessarily an easy task. There is no single command that will readily display all uses of an identifier in the ACLs; you must depend on your knowledge of the purposes of identifiers or you can write a specialized program.) While the ACL will still work when there are no holders of the identifier, in general you should strive to have a timely rights database and ACLs that are in step with it.

For example, suppose the summer students have left Treetown State and the fall semester is about to begin. Russ Ironwood, the security manager, might do a cleanup on the files that have ACLs that specify the identifier 84SUMMER. He would want to remove the identifier 84SUMMER from any ACE where it occurred before he removes the identifier from the rights database. Assume he discovers just one file, 84SUMLABEX.EXE with the following ACE:

```
(IDENTIFIER=84SUMMER,ACCESS=READ+WRITE+EXECUTE)
```

He invokes the VAX/VMS ACL Editor and obtains the following display of the ACL:

```
$ EDIT/ACL 84SUMLABEX.EXE
(IDENTIFIER=[BOTSTAFF,RDOGWOOD],ACCESS=READ+WRITE+EXECUTE+DELETE+CONTROL)
(IDENTIFIER=[BIOSTAFF,KMAHOGANY],ACCESS=READ+WRITE+EXECUTE+DELETE)
(IDENTIFIER=84SUMMER,ACCESS=READ+WRITE+EXECUTE)
(IDENTIFIER=)
```

Russ positions the cursor at the third ACE. By pressing the **PF4** key on the keypad, he deletes the line. He then presses **CTRL/Z** and exits the VAX/VMS ACL Editor, successfully updating it, as shown below:

```
$ EDIT/ACL 84SUMLABEX.EXE
(IDENTIFIER=[BOTSTAFF,RDOGWOOD],ACCESS=READ+WRITE+EXECUTE+DELETE+CONTROL)
(IDENTIFIER=[BIOSTAFF,KMAHOGANY],ACCESS=READ+WRITE+EXECUTE+DELETE)
(IDENTIFIER=)
CTRL/Z
%ACLEDIT-S-ACLUPDATED, ACL successfully updated
$
```

Next, Russ would remove the identifier 84SUMMER with the AUTHORIZE command REMOVE/IDENTIFIER. Note that if Russ were to reverse his procedure so that he issued the AUTHORIZE command REMOVE/IDENTIFIER before he looked at the ACL, he would find the third ACE took on an appearance that resembled the following:

```
(IDENTIFIER=%X80080011,ACCESS=READ+WRITE+EXECUTE)
```

Whenever you see an identifier in hexadecimal format in an ACE, you know that a general identifier has been deleted. Similarly, if you see an identifier displayed as a numeric UIC, the original identifier was a UIC that has been removed. Your best policy is to delete ACEs with numeric UIC or hexadecimal identifiers, so that you avoid the potential problem described next.

Security managers should be very careful not to reuse UICs too soon. That is, if an employee leaves and then you add a new user, avoid reassigning the old employee's UIC to the new employee. If you do, you run the risk that the new employee will gain some or all of the access rights of the old employee through lingering ACL entries that still reference the old UIC numerically.



Consider a different possibility. Suppose Russ knew that the identifier 84SUMMER would serve the same purpose the next year. Then he might choose to rename the identifier to 85SUMMER, instead. The following AUTHORIZE command would accomplish the renaming function:

```
UAF> RENAME/IDENTIFIER 84SUMMER 85SUMMER
identifier 84SUMMER renamed.
```

However, the renaming has an effect on existing ACLs—those that had ACEs that specified identifiers of 84SUMMER would now show 85SUMMER. Thus, before taking this step, Russ would have studied the ACL situation thoroughly to ensure that this change would be desirable in all cases.

### 5.2.5.4 Displaying the Rights Database

It is important that security managers regularly display the rights database and scrutinize it for both correctness and currentness. Two AUTHORIZE commands are used for this: SHOW/IDENTIFIER and SHOW/RIGHTS. For example, to display the identifier names, values, and attributes for all the users at Rainbow Paint, by username, Olive would issue the following command:

```
UAF> SHOW/RIGHTS/USER==*
Identifier      Value      Attributes
Identifiers held by LWHITE :
  ACCTSRCV      %X80080027 NORESOURCE
  PAYROLL       %X80080024 NORESOURCE
Identifiers held by OGREEN :
  PAYROLL       %X80080025 NORESOURCE
Identifiers held by RSILVER :
  PAYROLL       %X80080012 NORESOURCE
UAF>
```

Olive might prefer instead to see the identifiers according to UICs. She would use the following command:

```
UAF> SHOW/RIGHTS/USER=[*,*]
Identifier      Value      Attributes
Identifiers held by OGREEN :
  PAYROLL       %X80080025 NORESOURCE
Identifiers held by LWHITE :
  ACCTSRCV      %X80080027 NORESOURCE
  PAYROLL       %X80080024 NORESOURCE
Identifiers held by RSILVER :
  PAYROLL       %X80080012 NORESOURCE
UAF>
```

Another common requirement for security managers is displaying the rights of a particular user. Suppose Olive wants to review Lily White's rights. She would use the following **AUTHORIZE** command:

```
UAF> SHOW/RIGHTS LWHITE
Identifier                Value                Attributes
Identifiers held by LWHITE :
  ACCTSRCV                %X80080027        NORESOURCE
  PAYROLL                 %X80080024        NORESOURCE
UAF>
```

If Olive is more concerned with learning the names of all the holders of an identifier, such as **PAYROLL**, she might request the display with the **AUTHORIZE** command **SHOW /IDENTIFIER/FULL**, as follows:

```
UAF> SHOW/IDENTIFIER/FULL PAYROLL
Name                Value                Attributes
PAYROLL            %X8001006        NORESOURCE
  Holder
  OGREEN           NORESOURCE
  RSILVER          NORESOURCE
  LWHITE           NORESOURCE
UAF>
```

---

### 5.2.6 Setting Protection and Ownership Defaults for Users

Sometimes a security manager knows that a user will be using a directory or files that demand special protection. The security manager must recognize which of a number of possible defaults will affect the user, and exert additional control where the default protection is deemed inadequate. Section 4.5 describes default protection in detail. To review a little, there are three possible areas where security managers could specify protection defaults that would affect the user. In the order of increasing influence, depending on their existence, they are:

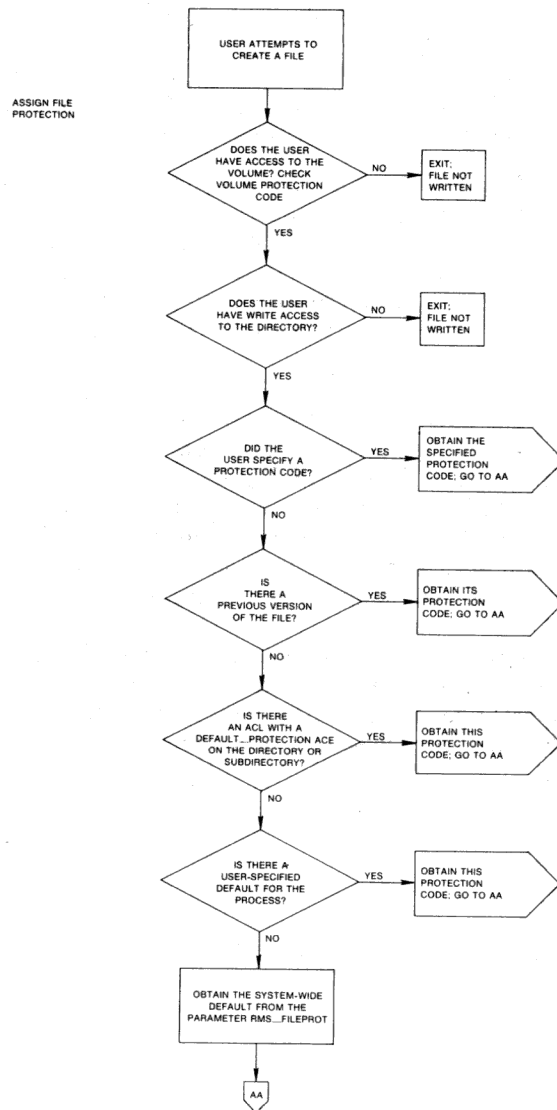
- The system-wide default for file protection that is provided by the **SYSGEN** parameter **RMS\_FILEPROT**. This value always exists and may be the only one to influence the outcome. Security managers can change the value of **RMS\_FILEPROT** with **SYSGEN**. However, the effectiveness of this value may be overridden by either of the remaining possibilities.

- The DCL command SET PROTECTION/DEFAULT (that may optionally appear in the user's own login command procedure) can specify the file protection placed on files that are created or modified by the user during the terminal session. At any time during a session, the user can also issue this command to override the value set by a previous SET PROTECTION/DEFAULT command. The SET PROTECTION/DEFAULT command negates the influence of the system-wide protection—for this user.
- The default protection for the specific directory can be specified in an ACL that is applied to the directory. If the DEFAULT\_PROTECTION ACE exists for the directory, all new files added to the directory, including subdirectories and their files, are subject to this protection code. This code overrides the system-wide default and the user-specified default (if any).

Security managers would also want to consider the protection optionally imposed on the volume through the DCL command SET VOLUME/PROTECTION. This protection code, if specified, could prevent a user from accessing any part of the volume, regardless of the protection code on the directory or the file. If no volume protection is specified with the SET VOLUME command, the volume will be open to all users.

In addition, as Section 4.4 demonstrates, the assignment of file ownership affects the outcome of any protection check. The operational effect of this combined protection structure can be depicted in a flowchart, as shown in Figure 5-5.

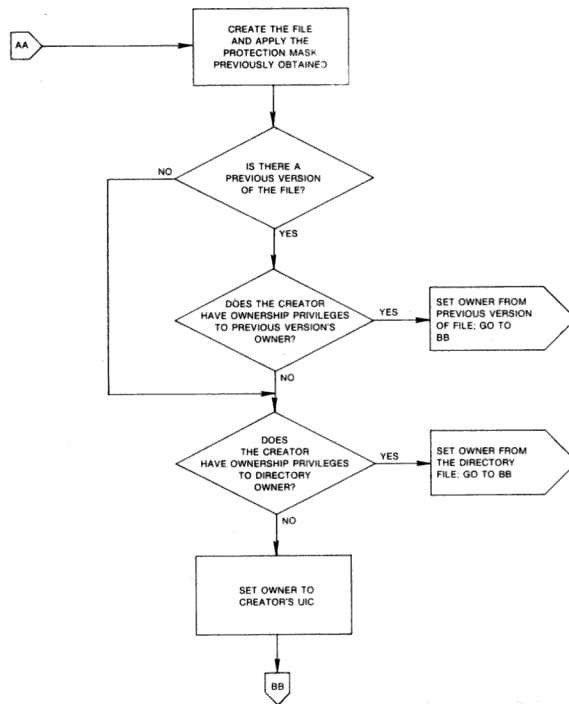
**Figure 5-5 Flowchart of File Creation**



ZP 2034/1-84

(Continued on next page)

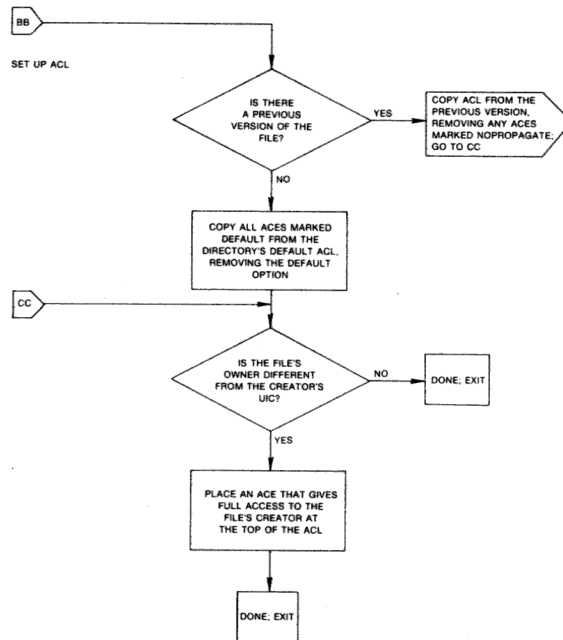
**Figure 5-5 (Cont.) Flowchart of File Creation**



ZK-2034/2-84

(Continued on next page)

**Figure 5-5 (Cont.) Flowchart of File Creation**



ZK-2034-3-84

### 5.2.6.1

#### Adjusting Protection Defaults

Given this understanding of default behavior, you may want to make some adjustments to control the default behavior in special cases. For example, unless the security manager invokes SYSGEN and changes it, the system-wide default protection code specified by RMS\_FILEPROT sets the user's default protection to:

(S:RWED,O:RWED,G:RE,W)

Assume that the volume protection has been set by the operator to:

(S:RWED,O:RWED,G:R,W).

## Security for the System Manager

The file protection on the directory [PROJECT] has been set to:

(S:RWED,O:RW,G:R,W).

Perhaps all the files that will be created in the subdirectory [PROJECT.DIARY] demand more stringent protection. The security manager, or any user who has the same access as the owner of the directory, could define a specific default protection code for this specific directory with an ACL consisting of just a DEFAULT\_PROTECTION ACE. For example, it might be preferable to set the protection code for all files created in the subdirectory to:

(S:RWED,O:RWED,G,W)

The following DCL command would provide the desired default protection:

```
$ SET ACL/ACL=(DEFAULT_PROTECTION,S:RWED,O:RWED) [PROJECT]DIARY.DIR
```

Once this ACL is placed on the directory file, all files created or modified in the directory will be subject to the default protection code. However, default protection codes should not induce a false sense of security. They are quickly overridden by any user with the BYPASS privilege, and can be circumvented under the right circumstances by users with SYSPRV, GRPPRV, or READALL privileges, as well. Furthermore, they are only defaults. The user who subsequently acquires CONTROL access to a file in the directory can include a specific protection code as a replacement for the default value on the file, with such DCL commands as:

- SET PROTECTION
- COPY/PROTECTION
- APPEND/PROTECTION

Generally, once the default protection code is replaced, the new code becomes the default and is propagated to subsequent versions of the file.

Another consideration is the proper use of the SET PROTECTION/DEFAULT command. If you provide a special login command procedure for your users, you may want to tailor the default process protection through this command. Such adjustment is beneficial—provided the user does not override it.

#### 5.2.6.2 Adjusting Ownership

To exert influence over the ownership of volumes, directories, or files, you can single out specific ones and apply the DCL commands SET VOLUME/OWNER, SET DIRECTORY/OWNER and SET FILE/OWNER, respectively, using your SYSPRV privilege. From Figure 5-5, the influence of the initial settings on the creation of new files is clear. However, one of the more interesting ways you can control the ownership of files, which influences not only the outcome of protection checks, but also the charging of disk space against disk quotas, is the use of the resource attribute with identifiers. This section explores that possibility in more detail.

To allow for more flexible management and accounting of disk space, identifiers can carry the optional resource attribute. This attribute, when present on an identifier, allows file space to be owned by and charged to that identifier. Thus, when files belong to a department or project, the space they consume can be charged to the appropriate department or project, rather than to the individual who created them. When users work on multiple projects, they can cause their disk space requirements to be charged to the related project rather than to their personal accounts.

#### Examples

Consider three users TOM, DICK, and HARRY who are members of the personnel department. A group directory [PERSONNEL\_DATA] has been set up for files related to the department's functions, and the space is to be charged to the department, rather than to the individual members. The owner identifier of the [PERSONNEL\_DATA] directory is PERSONNEL. TOM, DICK, and HARRY are all listed as holders of the identifier PERSONNEL in the rights database, giving them access to the files (presumably through suitable access control list entries). TOM and DICK are senior members of the department, and hold the PERSONNEL identifier with the RESOURCE attribute. Harry, as a junior member, does not have the attribute. A listing of Harry's rights would show the NORESOURCE attribute.

When TOM and DICK create files in their personal directories, the files are assigned their respective UICs as owner, to reflect the actual ownership of their directories. However, when either TOM or DICK create files in the [PERSONNEL\_DATA]



directory, these files obtain the file owner of PERSONNEL for two reasons:

- 1 The parent directory owner is PERSONNEL
- 2 The file creator possesses the identifier PERSONNEL with the RESOURCE attribute

(See the file ownership defaulting rules in Section 4.4.5.)

However, if TOM has write access to DICK's directory and creates a file in it, the file owner is TOM, for the following two reasons:

- 1 DICK's directory is owned by DICK
- 2 TOM does not hold the identifier DICK

On the other hand, user HARRY has not been entrusted to charge space to the PERSONNEL project. Therefore, when HARRY writes files into the [PERSONNEL\_DATA] directory, the owner is HARRY. Although HARRY holds the PERSONNEL identifier, HARRY lacks the RESOURCE attribute for that identifier.

As another example, consider FRED, who works as a consultant for both the biology and chemistry departments at a university. To charge file usage properly, the security manager has set up the project identifiers BIOLOGY and CHEMISTRY, and granted both to FRED with the RESOURCE attribute. FRED sets up subdirectories [FRED.BIOLOGY] and [FRED.CHEMISTRY], each owned by the corresponding identifier. FRED can specify the owner identifier with either the DCL command CREATE/DIRECTORY/OWNER\_UIC when creating them, or afterwards with the DCL command SET DIRECTORY /OWNER\_UIC. Thereafter, files that FRED writes into the [FRED.CHEMISTRY] directory receive the owner identifier CHEMISTRY, and so on. Files that FRED writes elsewhere receive the owner UIC of FRED (assuming the parent directory is not owned by CHEMISTRY or BIOLOGY).

FRED can also explicitly change the ownership of files that he has cataloged elsewhere with the DCL command SET FILE /OWNER, thus explicitly charging files to the appropriate project even though they are listed in some other directory. New versions of these files will inherit the same ownership, according to the ownership rules described in Section 4.4.

## Setting Up a Project Account

The following steps are necessary to actually set up a project identifier and directory:

- 1 Create the project identifier with the resource attribute and grant it to the appropriate individuals. For example, to create the project identifier of KITE\_FLYING with the resource attribute and to assign it to users GEORGE and LINDORF so that they can charge file space to it, you would use the following commands:

```
$ RUN SYS$SYSTEM:AUTHORIZE
UAF> ADD/IDENTIFIER KITE_FLYING /ATTRIBUTES=RESOURCE
{message}
UAF> GRANT/IDENTIFIER KITE_FLYING GEORGE /ATTRIBUTES=RESOURCE
{message}
UAF> GRANT/IDENTIFIER KITE_FLYING LINDORF/ATTRIBUTES=RESOURCE
{message}
```

- 2 Create the disk quota authorization for the project identifier. For example, the following command invokes the VAX/VMS Disk Quota Utility and assigns the identifier KITE\_FLYING 2000 blocks of disk quota with 200 blocks of overdraft:

```
$ RUN SYS$SYSTEM:DISKQUOTA
DISKQ> ADD KITE_FLYING /PERMQUOTA=2000 /OVERDRAFT=200
```

- 3 Create the project directory. For example, the following DCL command creates the project directory [KITE\_FLYING] and establishes the identifier KITE\_FLYING as the owner:

```
$ CREATE/DIRECTORY [KITE_FLYING] /OWNER=[KITE_FLYING]
```

- 4 Set up the necessary ACL and default ACL on the project directory. For example, the following DCL command places an ACL on the directory [KITE\_FLYING] that permits any holder of the identifier KITE\_FLYING to gain READ, WRITE, or EXECUTE access to the directory. It also ensures that files created in the directory will receive the same ACE as a default:

```
$ SET DIRECTORY [KITE_FLYING] /ACL= (-
$_ (IDENTIFIER=KITE_FLYING,ACCESS=READ+WRITE+EXECUTE),-
$_ (IDENTIFIER=KITE_FLYING,OPTIONS=DEFAULT,-
$_ ACCESS=READ+WRITE+EXECUTE))
```

Access must be granted through ACL entries, since the owner identifier of the directory and the files (KITE\_FLYING) does not match the UIC of any of the project members; thus, only WORLD access is available through the UIC-based protection mask. The first ACE of the specified ACL gives all project members READ and WRITE access to the directory; the second ACE gives them READ, WRITE, and EXECUTE access for all files created in the directory.

Note that project members are not allowed to delete (or control) files created by others. However, the members each have complete access to files that they have created in the directory, because the file system supplies an additional default ACL entry that grants full access to the creator. This ACE only appears when the owner of the created file does not match the UIC of the creator (and therefore the OWNER field of the UIC-based protection mask is not applicable).

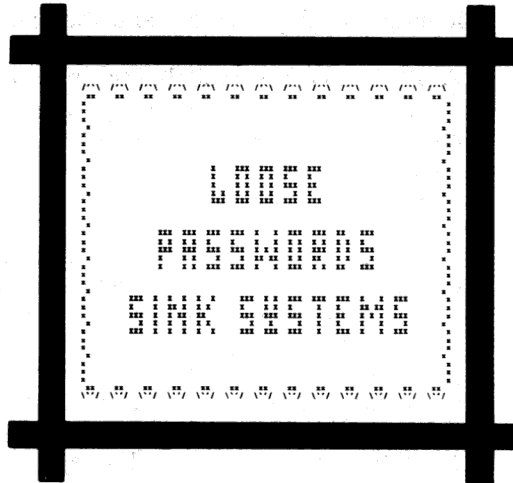
Thus, when LINDORF creates the file [KITE\_FLYING]THURSDAY.TXT, it receives by default the following access control list:

```
(IDENTIFIER=LINDORF,OPTIONS=NOPROPAGATE,  
ACCESS=READ+WRITE+EXECUTE+DELETE+CONTROL)  
(IDENTIFIER=KITE_FLYING,ACCESS=READ+WRITE+EXECUTE)
```

---

### 5.2.7 Password Management

Proper password management is your first line of defense on any system. While open systems will not employ passwords, the average commercial site will typically find them desirable, and the maximum security site will find them indispensable. In fact, the maximum security site may choose to impose a double password scheme through the use of primary and secondary passwords. Furthermore, the maximum security site is likely to implement system passwords as well, to restrict access to certain terminals to those who know the system password.



ZK-2072-84

#### 5.2.7.1

#### Initial Passwords

When you open an account for a new user with AUTHORIZE, you must give the user a username and an initial password. When you assign temporary initial passwords, try to observe all the guidelines recommended to the users in Section 3.1.3.10. Set a good example. One way to do this is to use the automatic password generator to derive some choices for passwords. That way you resist the temptation to use first names or items that would have a suggestive pattern, such as names of towns, names of departments, breeds of dogs, authors of books, and so forth. Remember, once your pattern becomes evident to the users, you make password guessing an intrigue, and you may create an unnecessary system exposure.

To take advantage of the automatic password generator while you are using the VAX/VMS Authorize Utility to open an account, enter the AUTHORIZE qualifier /GENERATE\_PASSWORD. The system responds by offering you a list of automatically generated password choices. You simply select one of these passwords and continue on with the remaining details of opening the account.

When you add a new user to the UAF, you may choose to define the user's password as previously expired, thereby forcing the user to change the initial password during the first login. Use the AUTHORIZE qualifier /PWDEXPIRED for this purpose. The system behaves just as if the password had reached its expiration date, as described in Section 5.2.7.5.

Pre-expired passwords are conspicuous in the UAF record listing. The entry for the date of the last password change carries the notation:

<pre-expired>

Whether or not you define initial passwords so that they have previously expired, it is very important to stress to users that they must change the initial password as soon as they login the first time. You can further facilitate this by including the first login as part of your user training. When you are physically present, you ensure both that little time has elapsed from account creation until first login, and that the user actually succeeds in changing the initial password.

---

### 5.2.7.2

#### Password Verification

Section 3.1.3 discussed the use of a one-way encryption algorithm by VAX/VMS to deter efforts to decode the stored passwords. The description of the method VAX/VMS uses to verify that a user submits the correct password, implies that only the precise same password can match. Actually, there is a miniscule possibility that two different passwords might generate the same encrypted value. Such an event would be called a *password collision*, but the probability of collision with the VAX/VMS one-way encryption algorithm is on the order of  $10^{-20}$ , which is virtually negligible. The length of the passwords has no effect on the likelihood of collisions. With the VAX/VMS one-way encryption algorithm, a 64-bit hashed value is always created, regardless of the length of the password that is input.

### 5.2.7.3

#### System Passwords

Section 3.1.3.1 introduces system passwords, which you might implement to control access to particular terminals. Sometimes system passwords are desirable simply as another level of security, but most often they are selected as a means of controlling access to terminals that might be targets for unauthorized use. Typical of such terminals would be any of the following:

- All terminals using dialups or public data networks for access
- Terminals on lines that are publicly accessible and not tightly secured such as computer laboratories at universities
- Terminals in more remote areas not frequently inspected
- Terminals intended for use only as spare devices
- Terminals the security manager wants to reserve for security operations

Note that the use of a system password is incompatible with the use of the DIGITAL-supplied terminal concentrator known as the LAT-11. For more information about the LAT-11, see the documentation provided with the LAT-11 software.

There is a general technique for implementing system passwords and a more specific one when remote logins are involved. Consider the general case first. Implementing system passwords is a two-stage operation involving the DCL commands SET TERMINAL and SET PASSWORD. First, you must decide which terminals require system passwords and then, for each terminal, you issue the DCL command: SET TERMINAL/SYSPWD/PERMANENT. You might decide to test your plans out temporarily this way. However, once you are satisfied that you have selected the right terminals, you would incorporate these commands in SYS\$MANAGER:SYSTARTUP.COM, so that the terminal setup work is done automatically at system startup time. You can remove the restriction on a terminal at any time. You simply invoke the DCL command SET TERMINAL/NOSYSPWD/PERMANENT for that terminal.

As your next step, choose a system password and implement it with the DCL command SET PASSWORD/SYSTEM, which requires the SECURITY privilege. This command will prompt you for the password and then ask you to reenter it for

verification, just as is done for user passwords. You can request automatic password generation by including the /GENERATE qualifier.

To enable the use of the system password for the remote class of logins (those accomplished through the DCL command SET HOST), set the appropriate bit in the default terminal characteristics parameter using SYSGEN. This is bit 19 (hexadecimal value 80000) in the parameter TTY\_DEFCHAR2. Note that if you set this bit, you must invoke the DCL command SET TERMINAL/NOSYSPWD /PERMANENT to disable system passwords for each terminal where you do not want the feature. (As before, consider placing the SET TERMINAL commands you have tested in SYS\$MANAGER:SYSTARTUP.COM.) Follow the steps in the preceding paragraph to set the system password.

In choosing a system password, follow the recommendations of Section 3.1.3.10. The most important steps are to choose a non-English string of characters and digits, with a length of 6 or more. Change the password frequently. Always change the system password as soon as a person who knows the password leaves. It is a good practice to share the system password only with those who need to know. A side benefit of maintaining a small circle of users who know the system password is that you avoid spending excessive time changing the password in response to employee turnover and so forth. (The system password is not subject to an expiration time.)

The system password is stored in a separate UAF record. This record is hidden from view; it cannot be displayed. However, you can modify the password with AUTHORIZE, if necessary. You may recall that the DCL command SET PASSWORD /SYSTEM (the normal means of setting and changing the system password) requires that you enter the old system password prior to changing it. Since it could become necessary to change the system password when no one is available who remembers what the current system password is, AUTHORIZE offers the means. For example, to change the system password to ABRACADABRA, you could invoke AUTHORIZE and simply enter the following command:

```
UAF> MODIFY/SYSTEM_PASSWORD=ABRACADABRA
```

System passwords certainly have a purpose, particularly in the high-security environment. Their primary function is to form a first line of defense for publicly accessible ports, to fend off visitors, and to prevent would-be intruders from even learning the identity of the system. However, requiring system passwords is deliberately confusing and downright unfriendly when authorized users are unaware that they are required on certain terminals. Such users may stray upon unused terminals and attempt logins with no success. Thus, to avoid false reports of defective terminals or systems, be sure you inform your users which of the terminals allocated for their use will require system passwords.

Another slight potential disadvantage of system passwords exists, but it is less commonly seen. Where system passwords are not applied to either control access through dialups or on publicly accessed lines, only a few people might know the system password. In that case, there is the possibility of encumbered operations if the personnel who know the password are unavailable, incapacitated, or happen to forget the password. However, this problem can be resolved quickly by anyone with SYSPRV privilege who can invoke AUTHORIZE to apply the command MODIFY/SYSTEM\_PASSWORD.

---

#### 5.2.7.4

#### Primary and Secondary Passwords

You must also give some thought at account initialization time as to whether or not to require primary and secondary passwords, as described in Section 3.1.3.7. The use of dual passwords is cumbersome and mainly warranted at sites with high-level security concerns. Dual passwords offer two advantages: when used on a widespread basis they facilitate the verification of the physical identity of each user at login time through visual contact, and when used in limited cases, they single out accounts that can only be logged in to when two persons are present.

Sites with security requirements in the medium range might want to reserve the dual password tactic as a special tool for application when there seem to be unexplained break-ins into one or more accounts, even after the password has been changed and the use of the password generator has been enforced. Select the problem accounts and make them a temporary target of this restriction. If the problem goes away when you institute personal verification through the secondary password, you know you have a personnel problem. Most likely, the authorized user is revealing the password for



the account to one or more other users who are abusing the account or revealing the password to other abusers.

You implement dual passwords with the AUTHORIZE qualifier /PASSWORD. For example, to impose dual passwords on an existing account, such as Rich Silver's, the security manager would invoke AUTHORIZE and issue the following command:

```
UAF> MODIFY RSILVER /PASSWORD=("", WRYGLOG)
user record(s) updated
```

This command does not affect the primary password that already exists for the account, but adds the requirement that a secondary password be provided at each subsequent login. The secondary password is given the initial value of WRYGLOG, and acquires the same password lifetime and minimum length values that are in effect for the primary password. Also, if the /FLAGS=GENPWD qualifier has been specified for this account, the secondary password can only be changed under the control of the automatic password generator.

**Note:** Secondary passwords can not be specified in access control strings for file accesses over the network. Therefore, you should not require secondary passwords on accounts that are likely to require network access. However, note the distinction between network and remote access. Secondary passwords are quite appropriate for accounts that may require remote access using the DCL command SET HOST.

### 5.2.7.5 Enforcing Minimum Password Standards

Security managers can also use AUTHORIZE to impose some minimum standards to individual users regarding their use of passwords. Specifically, qualifiers provided by AUTHORIZE control the minimum length of passwords and how soon the passwords will expire.

#### Password Expiration

With the AUTHORIZE qualifier /PWDLIFETIME, you can establish the maximum length of time that can elapse between password changes before the user will be forced to change the password or lose access to the account. By default, the value of /PWDLIFETIME is six months. You can change the frequency requirements for user password changes by specifying a different delta time value for the qualifier. For example, to

require a user to change the password every two months, you would specify the qualifier as `/PWDLIFETIME=60-0`.

The `/PWDLIFETIME` qualifier applies to both primary and secondary user passwords, but not to the system password. That is, each primary and secondary password for a user is subject to the same maximum lifetime. However, the passwords can change at separate times. As soon as the user completes a password change, that individual password's clock is reset; the new password value can exist unchanged for the length of time dictated by `/PWDLIFETIME`.

The Authorize Utility also provides two flags related to primary and secondary password expiration. These flags, `PWD_EXPIRED` and `PWD2_EXPIRED`, can be specified with the `/FLAGS` qualifier. The first flag, `PWD_EXPIRED`, is set on after the primary password expires and the user has had one last chance to change the password and failed to do so. The second flag, `PWD2_EXPIRED`, is set on after the secondary password expires and the user has had one last chance to change the secondary password and failed to do so. If either `PWD_EXPIRED` or `PWD2_EXPIRED` is set, the account will be disabled for logins, since the user failed to employ the last chance to change the password during the last login.

As soon as the user successfully changes the password, VAX/VMS resets the flags as appropriate. That is, the flag `PWD_EXPIRED` becomes `NOPWD_EXPIRED` as soon as the primary password is changed. Similarly, the flag `PWD2_EXPIRED` becomes `NOPWD2_EXPIRED` as soon as the secondary password is changed. As security manager, you can invoke `AUTHORIZE` and reset the flags. This might prove to be a convenient way, for example, to give the user another chance to reset the password.

The use of a password lifetime forces the user to change the password with regular frequency. You must apply some judgement regarding reasonable password lifetimes at your site. The lifetime can be different for different users. Users with access to critical files generally should have the shortest password lifetimes compared to your other users.

System passwords have an unlimited lifetime. Therefore, you should be rigorous about changing the system password whenever the situation warrants. For example, system password changes are warranted when personnel leave who

had access to the system password. In general, it is unwise for any password to go unchanged longer than six months.

### Minimum Password Length

With the AUTHORIZE qualifier /PWDMINIMUM, you can direct that all choices that the user makes for passwords must be at least a certain number of characters long. Since this is a minimum length, the users can still specify passwords up to the maximum length of 31 characters.

This length applies equally to primary and secondary passwords. This value is only enforced through the execution of the DCL command SET PASSWORD. Thus, as security manager, you may specify initial passwords through AUTHORIZE that are shorter than the minimum. However, doing so may confuse your users unnecessarily. Furthermore, initial passwords inherently introduce security weaknesses. By selecting short initial passwords, you compound the problem. Generally, it is good practice to observe the same rules you expect your users to follow.

There is always a minimum password length in effect for each user. It is either the default of 6 or another value established by the /PWDMINIMUM qualifier. Thus, if the user specifies the DCL command SET PASSWORD/GENERATE=n, n must be a value at least as great as the minimum value in effect. If the number n is less than the current minimum enforced in the UAF, it is simply disregarded; no message appears. However, the list of five password choices that VAX/VMS generates for the user complies with the current minimum password length.

The password generator creates passwords that range in length between n and n+2, where n is the specified or minimum length. In addition, the maximum values for n and n+2 that the password generator can accommodate are 10 and 12, respectively. Longer passwords require an inordinate amount of CPU time to generate.

The system password is not subject to a minimum length. Common sense dictates that the guidelines that apply to user passwords are equally applicable to system passwords. Thus, you should normally choose system passwords that are 6 to 10 characters long.

#### 5.2.7.6

#### Requiring the Password Generator

The `/FLAGS=GENPWD` qualifier in `AUTHORIZE` allows you to direct that a user must use the automatic password generator when changing a password. At some sites, all accounts will be created with this qualifier. At other sites, the security manager may be more selective.

If you find yourself considering which users are candidates for this requirement, your criteria should be whether or not the user will have access to sensitive data that must not be compromised by a break-in and/or whether or not you sense the user will be sloppy. Users with access to sensitive data are prime candidates for this restriction, as are sloppy users.

If your policy is to request that users voluntarily apply the password generator, you have to reconsider from time-to-time just how willing and compliant your users seem to be. If you sense that users are not using the password generator on their own, you have at least two choices. You may decide to force compliance by adding the `/FLAGS=GENPWD` qualifier to most or all user accounts. Another option is to take the responsibility on yourself to enforce sensible passwords. You can accomplish this with the `AUTHORIZE` qualifier `/FLAGS=LOCKPWD`. In this way, you take the ability to change passwords away from the users and assume the responsibility yourself.

There is no easy way to know if users are voluntarily using the password generator, but if you are observant, you will notice other kinds of more obvious infractions of your rules, such as failing to log out before leaving a terminal and so forth. As soon as you become uncomfortable about your users' good intentions or ability to follow through, it is time to crack down. Requiring frequent password changes with suitable passwords is an important area to address in any security program.

#### 5.2.7.7 Protecting Passwords

Just as users have responsibilities to protect passwords, so do security managers. In addition to all the recommendations included in Section 3.1.3.10, the responsible security manager must observe the following guidelines:

- Make certain the passwords on the standard accounts SYSTEM, FIELD, and SYSTEST are properly managed, either by disabling the accounts with the AUTHORIZE qualifier /FLAGS=DISUSER when they are not in use or by changing the passwords to obscure values and managing the passwords properly. The *Guide to VAX/VMS System Management and Daily Operations* offers some pertinent suggestions.
- Do not permit an outside or in-house service organization to dictate the password for an account that they use to service your system. Such service groups tend to use the same password on all systems and their accounts are usually privileged. The best policy on seldom-used accounts is to set the AUTHORIZE flag DISUSER, and then only enable the account when it is needed. Alternatively, you can change the password immediately after each use, and notify the service group of the new password when they need it next.
- Dispose of accounts no longer in use.
- If you have an account on a system that stores the passwords in plaintext, be sure to choose a different password everywhere else you have an account. In this way, if your less protected password is discovered, you have not disclosed the password on other accounts as well.
- Do not produce listings of the accounts with the VAX/VMS Authorize Utility commands SHOW or LIST and then leave those listings lying around where they could be read or stolen. Such carelessness would undermine the usual efforts to keep the user authorization files protected, as described next.
- Maintain adequate protection of the user authorization files. Note that the user authorization files (SYSUAF.DAT and NETUAF.DAT), are owned by group 1, which should be the system account. There should be no other users in this group. Accordingly, the categories SYSTEM, OWNER and GROUP are synonymous. Normally these authorization files need only be protected to disallow access to the world category, as done through the default protection supplied by

DIGITAL. Generally, you need not enhance the default UIC-based file protection with an ACL. (You might, however, use ACLs on these files or their listing files (SYSUAF.LIS and NETUAF.LIS) to grant access to selected individuals or to establish file access auditing to the files.

Other actions that are not strictly forms of password protection but either reduce the potential of password detection or limit the extent of the damage if passwords are discovered or bypassed, follow:

- Generally avoid giving multiple users access to the same account.
- Do not leave the dialup telephone numbers where they can be found and tempt users to try to discover passwords over dialup lines.
- If your system has accounts available to outsiders, such as HOST accounts or Quality Assurance Reporting accounts, be that certain these accounts are captive accounts that are contained by captive command procedures. (See Section 5.8 for more information.)
- All accounts that do not require a password should be captive accounts, as well.
- Extend privileges to users with extreme caution. Section 5.3.6 will clarify that inside users with the special privileges (such as BYPASS, SYSNM, SETPRV, READALL, and GRPPRV) and UICs with low group numbers will not always need other users' passwords to get into parts of their accounts. With one or two of these privileges and some cleverness, a user can circumvent a great deal of the effectiveness of password control.
- Protect your own files, using all the techniques recommended in Section 4.8.
- Ensure that the files containing components of the operating system are adequately protected. (See Appendix C.)
- Use the AUTHORIZE qualifiers /NOINTERACTIVE and /NOBATCH when you set up proxy login accounts to permit file access from other nodes—so that the accounts are truly network login accounts.

---

## 5.2.8 Login Options

As Section 3.1.2 indicates, you can control the display of various pieces of information that appears by default at login time. You must consider the advisability of displaying announcement, welcome, last login, and new mail messages. Each have certain merits, but also implications for tight security systems. In addition, this section discusses the use of the secure server and how to set up breakin detection and evasion.

---

### 5.2.8.1 Controlling the Announcement Message

The system logical name SYS\$ANNOUNCE must be defined to provide an announcement message. The *Guide to VAX/VMS System Management and Daily Operations* describes the proper technique to use. The announcement message, if it exists, appears as soon as the user attempts to login. That is, the system displays SYS\$ANNOUNCE as soon as it has successfully autobauded the terminal (if autobauding is enabled) and received the system password (if system passwords are enabled).

The definition you provide here affects all users on the entire system. Since this message may provide a clue to the identity of the operating system, you may decide to dispense with it.

---

### 5.2.8.2 Controlling Disconnected Jobs

There are several ways to disable the disconnected job feature, if you become concerned about the resources it consumes. However, consider which techniques will affect the system-at-large and the full ramifications of disabling before you adopt a particular approach.

Two of your options work at the terminal or user level. To affect particular terminals, you can use the DCL command SET TERMINAL/PERMANENT/DISCONNECT. You can also set the AUTHORIZE qualifier /FLAGS=DISRECONNECT for particular users. (An applications account used by multiple users is a good candidate for the DISRECONNECT flag, to prevent the users from connecting to each other's jobs.) At the system level, you can disable virtual terminals on a system-wide basis with the SYSGEN parameter TTY\_DEFCHAR2. However, this has other effects as well. You can also set the amount of time allowed for reconnection to a much smaller value than the default of 15 minutes. (You use the SYSGEN

parameter TTY\_TIMEOUT for this purpose, and the result also affects the system at large.) Limiting the connection time tends to minimize the number of users who receive messages, but it also affects the usefulness of the connection feature.

---

### 5.2.8.3

#### Controlling the Welcome Message

You have two options regarding the welcome message that is normally transmitted when the system accepts a valid user password. Similar to the announcement message, the welcome message is controlled through a system logical name, which is SYS\$WELCOME. If you do not define SYS\$WELCOME, a standard welcome message is provided for all users. This welcome message reveals the operating system and version number, as well as the node if SYS\$NODE is defined.

If you prefer to write your own message, you can define some other message for SYS\$WELCOME. Or using the same technique, you might place the following DCL command in SYS\$MANAGER:SYSTARTUP.COM, so that a blank line is printed:

```
$ DEFINE/SYSTEM SYS$WELCOME " "
```

(See the *Guide to VAX/VMS System Management and Daily Operations* for details.)

If you prefer to selectively disable the message for individual users, you can leave SYS\$WELCOME alone and use the AUTHORIZE qualifier /FLAGS=DISWELCOME on individual UAF records.

---

### 5.2.8.4

#### Controlling the Last Login Messages

You can selectively disable the appearance of the series of three possible messages that detail information regarding the last logins and number of failed attempts at logging in. You choose the /FLAGS=DISREPORT qualifier provided with AUTHORIZE and enter it for specific users.



#### 5.2.8.5 Controlling New Mail Announcements

You can prevent users from receiving a notification that they received new mail since the last time they were logged in, by specifying the /FLAGS=DISNEWMAIL qualifier with AUTHORIZE. This would be primarily a user convenience since the action has no particular security implications in itself. However, from an operational standpoint, it is sensible that if a user cannot invoke the VAX/VMS Mail Utility, there is no reason to tantalize that user with messages indicating that some mail has been sent. Thus, when you decide to prohibit mail access for a user, you might want to disable the new mail message at the same time. You can accomplish your dual purpose by specifying the following AUTHORIZE qualifier:

```
/FLAGS=(DISMAIL,DISNEWMAIL)
```

This is particularly appropriate for captive accounts.

#### 5.2.8.6 Controlling the Number of Retries on Dialups

You can control the number of retry attempts the user is allowed for login attempts through dialups. In this way, if the user makes a harmless typo or two after obtaining the connection, the user will not automatically lose the connection. Instead, the user receives a grace period and a limited number of attempts to succeed. This is an important aid to your users who may be unskillful typists. At the same time, it is a deterrent to the troublesome user who dials in with a known username and attempts to achieve a login by using a small computer programmed to enter every word in the dictionary as a password.

To implement control of retries, you use two of the *LGI parameters* provided with SYSGEN (see the *VAX/VMS Utilities Reference Volume*). These parameters are LGL\_RETRY\_TMO and LGL\_RETRY\_LIM. If you do not change the parameters, the default values afford the users three retries with a 20-second interval between each. This means that users will only lose the connection if they either fail to specify a valid password within three tries, or they spend more than 20 seconds between two of their tries.

Note that these values apply to every user on the system who is permitted to access the system through a dialup line.

The following example illustrates how you might set the total number of retry attempts to six, allowing a half-minute interval between tries. Since these LGI parameters are dynamic, you could change them and test them out before performing the SYSGEN command WRITE CURRENT and rebooting the system.

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> SET LGI_RETRY_LIM 6
SYSGEN> SET LGI_RETRY_TMO 30
SYSGEN> WRITE ACTIVE
{OPCOM messages show modification has been made}
SYSGEN> EXIT
$
```

### 5.2.8.7

#### Controlling Breakin Detection and Evasion

Section 5.2.8.6 shows how to control the number of login retries for users dialing in. By limiting the number of retries to a reasonable number on each dialup login, you make the job of dialing up and trying out every password combination more difficult for probing outsiders. But this is insufficient to completely evade break-ins. First of all, the user attempting this technique is determined; a small obstacle like redialing is not going to prove an effective deterrent. Secondly, this technique only applies to dialups. And remember, some outsiders will acquire electronic dialers to bypass the tedium of redialing. Also, outsiders willing to expend this much effort into breaking in may even put some computer processing power behind their attempts.

VAX/VMS offers additional methods of discouraging breakin attempts. These methods also use SYSGEN parameters in the LGI category. One of the parameters (LGL\_BRK\_LIM) defines a threshold count for login failures that triggers suspicion that a breakin attempt is in progress. That is, when the count of login failures exceeds the LGL\_BRK\_LIM value, within a reasonable time interval, the system assumes a break-in is in progress. Only login failures caused by specifying invalid passwords are counted, and they must be from a specific source. That source can be any of the following combinations:

- A specific terminal and a specific **valid** username. This dual association is the default. However, as described below, you have the option of counting failures on usernames only. Note also that invalid user names are all counted together as a single class per terminal. This count of invalid usernames is irrelevant to locking an intruder out, since an invalid

username will never log the intruder in. However, the count of invalid usernames is used to trigger security alarms. See Section 5.9.

- A specific remote node and a specific remote username.
- The username of the creator of a detached process.

By default, LGL\_BRK\_LIM will permit 5 failed login attempts from one of these sources. (Security managers can adjust the value of LGL\_BRK\_LIM with SYSGEN.)

The SYSGEN parameter LGL\_BRK\_TERM controls the association of terminals and usernames for purposes of counting failures. By default, VAX/VMS sets this parameter to 1 so that they are tracked together. If you set this parameter to 0, the terminal is not included in the association; the failures associate on username only. This feature is useful if you use terminal concentrator switches, LAT-11, or similar facilities, in which the terminal name that VAX/VMS Login sees is not a good indication of the identity of the actual terminal.

Another key parameter, LGL\_BRK\_TMO, controls how long VAX/VMS remembers a login failure. The initial failure on each source is given an expiration time that represents the current time plus the delta time given by LGL\_BRK\_TMO. Each additional failure on that source adds another delta of LGL\_BRK\_TMO to that entry, thus extending the length of time before the login failures are forgiven. The cumulative effect is that the more failures made by a source, the greater the window of time in which additional failures will count toward the critical number defined by LGL\_BRK\_LIM. If no more failures occur by the time the expiration point is reached, all failures are forgiven for that source. Note, however, that the failure count is **not** reset by a successful login.

For example, assume the default values are in effect. LGL\_BRK\_LIM specifies no more than 5 login failures from one source. LGL\_BRK\_TMO is set for 5 minutes. Assume that an outsider starts firing usernames and passwords at the system. When the first password fails, the clock starts to run and the user has 4 more tries in the next 5 minutes. When the second attempt fails about 30 seconds thereafter, the user has 3 tries left that will be counted over the next 9.5 minutes. When the third attempt fails 30 seconds later, the login failure observation time now extends out to 14 minutes. The fourth failure occurs about 1 minute later and the fifth failure occurs within another

30 seconds. By this time the observation time has reached 22.5 minutes. As a result, this source cannot afford another failure for a whole 22.5 minutes without triggering evasive action.

The effect is that the system tolerates an average rate of login failures that is the reciprocal of the parameter `LGL_BRK_TMO`. For example, if the default value of `LGL_BRK_TMO` (which is 300 seconds or 5 minutes) is in effect, the average rate of tolerable login failures is 1 every 5 minutes. When the rate of login failures exceeds the tolerable rate and the critical number of 5 failures is reached (the default value of `LGL_BRK_LIM`), the system concludes a break-in is in progress that calls for evasive action.

The evasive action consists of not accepting any logins from the offending source for a period of time. In the case of a terminal as the source (when `LGL_BRK_TERM` equals 1), no one can log in at that terminal with the username that is under suspicion for a period of time. (However, other users may log in from that terminal.) In the case of a source on a network, the specific remote user at that node is prohibited from logging in from that node for a period of time. Consequently, login attempts that provide valid username and password combinations that should otherwise succeed, now are rejected during this interval, but only from the now-presumed intruder at that source. Once the interval elapses, the rules return to normal. As a result of this form of evasive action, however, outsiders are less likely to succeed in learning the correct password by using repetitive trial-and-error attempts.

The duration of the evasive action is influenced by the `LGL_HID_TIM` parameter. The actual time depends on an additional random number (in the range of 1 to 1.5) that is used as a multiplier. The product of `LGL_HID_TIM` and the random number yields the actual duration of evasive action. The formula could be represented as:

$$\text{Evasion time} = \text{LGI\_HID\_TIM} * (\text{random number})$$

The inclusion of a random amount of time helps to obscure the true evasion time somewhat. An outsider who learned the value of `LGL_HID_TIM`, could not be assured that the evasive action would persist for exactly that length of time. However, too unpredictable a result would introduce unnecessary problems for security managers.

These parameters affect all terminals, users, and nodes that access the system. Since they are dynamic, you can reset them without rebooting the system.

Several loopholes exist here for the determined outsider. If the values of `LGL_BRK_LIM` and `LGL_BRK_TMO` can be learned or guessed, the outsider modifies the technique to perform it over sufficiently long intervals that suspicion is not triggered, or else changes terminals, nodes, and/or usernames frequently enough to avoid detection. Certainly the problem has become more complex for the outsider. However, security managers should not develop a false sense of security that simply because they have implemented breakin detection they have completely removed this potential problem.

The technique of counting failures per terminal and username raises the system's exposure somewhat, because the password guess rate for a particular username is multiplied by the number of available terminals. The benefit of this approach, however, is that it sharply reduces the denial of service problem that could result from simply counting failures per terminal or per username. If VAX/VMS had adopted the latter approach, a malicious user could put an entire terminal room (or a user's account) completely out of service for a period of time with very little effort.

By setting `LGL_BRK_TERM` to 0, you can shift this balance in favor of detecting attempts more promptly, at the expense of increasing the risk of denial of service to legitimate users.

The `SYSGEN` parameter `LGL_BRK_DISUSER` provides you the option of making the effects of breakin detection more severe. If you set this parameter to 1, VAX/VMS sets the `DISUSER` flag in the `UAF` record for the account where the break-in was attempted. Thus, that username is disabled until you manually intervene. However, the service denial effects of this option can be very severe. A malicious user can easily put all the known accounts out of service in a short time—including yours! To recover, you must log in on the system console, where the `SYSTEM` account is **always** allowed to log in. VAX/VMS stores, in the breakin database, information about login failures that originate from a specific source. You can use the `DCL` command `SHOW INTRUSION` to display the contents of the breakin database, and the `DELETE/INTRUSION_RECORD` command to remove entries from the breakin database. See the

*VAX/VMS DCL Dictionary* for additional information on these commands. Entries in the breakin database have the following format:

Intrusion	Type	Count	Expiration	Source
-----------	------	-------	------------	--------

The information provided in the fields in each entry is as follows:

Intrusion	Class of intrusion.
Type	Severity of intrusion as defined by the threshold count for login failures. The SYSGEN parameter, LGI_BRK_LIM, defines the threshold count.
Count	Number of login failures associated with a particular source.
Expiration	Absolute time when VAX/VMS stops keeping track of login failure. The SYSGEN parameter, LGI_BRK_TMO, controls this time.
Source	Origin of the login failure.

The information in the breakin database is controlled by the SYSGEN parameters in the LGI category.

Chapter 6 further discusses how to know your system is under attack and appropriate counteractions.

### 5.2.8.8

#### Using the Secure Server

Section 3.1.3.8 describes password grabbers—that class of programs designed to steal passwords from unsuspecting users who log in to turned on terminals. To combat this particular problem, VAX/VMS provides a secure terminal server that the security manager can invoke. The purpose of the secure server is to stop any currently executing process prior to the start of a login at that terminal.

You invoke the secure server on a terminal-by-terminal basis, with the following DCL command:

```
SET TERMINAL/PERMANENT/SECURE/DISCONNECT
```

Once you set the terminal up in this fashion, the user must press the BREAK key followed by the RETURN key to initiate a login. The login proceeds as usual.

You may or may not want to apply this constraint to all terminals. On the one hand, you make the login procedure consistent throughout the site and avoid some possible confusion. On the other hand, certain applications that may use the terminal as a communications line may need to use the BREAK key for their own purposes. This would be incompatible with the secure terminal server.

The secure server feature is also incompatible with autobaud handling. This is not really a problem, because autobauding is only necessary on modem terminals; that is, on switched or dialup terminals. The modem handling on such terminals performs the equivalent of secure server functions. For secure operation, set up the terminal characteristics as follows:

- For local terminals (direct wired) use the following SET TERMINAL qualifiers:  
`/NOMODEM/SECURE/DISCONNECT/NOAUTOBAUD`
- For switched terminals (data switch, dialup, and so forth), use the following SET TERMINAL qualifiers:  
`/MODEM/AUTOBAUD/NOSECURE/DISCONNECT`

Specify /DIALUP if the terminal port is accessible through a telephone line or the equivalent, regardless of the path (direct modem, data switch, concentrator, public data network, and so forth).





Always specify /DISCONNECT to ensure against password grabbers. (To prevent disconnected jobs from filling up your system, set the SYSGEN parameter TTY\_TIMEOUT to a small timeout value, which hastens their automatic removal.)

If you decide to be selective, the best candidates for inclusion of the /SECURE qualifier are directly wired terminals located in public areas or remote, unsecured areas. Terminals never used for local or dialup logins are not subject to this problem. Terminals that are closely supervised during logins may also not require this measure. (Remember to follow the suggestion in Section 5.2.7.3 and put the SET TERMINAL commands in SYS\$MANAGER:SYSTARTUP.COM.)

---

### 5.2.9 Using the Automatic Login Facility

You can assign accounts to particular terminals to enable an *automatic login* feature. This feature permits users to login without specifying a username. VAX/VMS associates the username with the terminal and maintains these assignments in the file SYS\$SYSTEM:SYSALF.DAT, commonly referred to as the *automatic login file* or ALF. You maintain this file with the command procedure SYS\$MANAGER:ALFMAINT.COM.

The ALF consists of one record for each terminal on which automatic logins are enabled. Each record consists of two fields: the device name of the terminal followed by the username of an account. The device names must be unique within the file. However, the same username can occur in any number of records; that is, one account can be automatically logged in to any number of terminals.

Use the ALFMAINT command procedure to maintain the ALF. The following commands invoke ALFMAINT for the system ALF:

```
$ SET DEFAULT SYS$MANAGER
$ @ALFMAINT
```

---

#### 5.2.9.1 Adding New Records

Respond to the "Terminal (ddcu)?" prompt with the physical name of a terminal (OPA0, TTAX, or TXAX). Respond to the "Username?" prompt with a valid username. If you enter an invalid terminal name, you receive an error message and are reprompted. The username is not checked for validity.

---

**5.2.9.2 Modifying Records**

Respond to the "Terminal (ddcu)?" prompt with the name of a terminal already in the ALF. Respond to the "Username?" prompt with the name of the new user to be associated with the terminal. Respond to the prompt "Do you want to change this record (Y/N)?" with the letter Y (uppercase or lowercase) to modify the record; any other response cancels the modification.

---

**5.2.9.3 Deleting Records**

Respond to the "Terminal (ddcu)?" prompt with the name of a terminal already in the ALF. Respond to the "User?" prompt by pressing the RETURN key. Respond to the "Do you want to delete this record (Y/N)?" prompt with the letter Y (uppercase or lowercase) to delete the record; any other response cancels the deletion.

---

**5.2.9.4 Exiting from ALFMAINT**

Respond to the "Terminal (ddcu)?" prompt by pressing the RETURN key or typing EXIT.

To access the ALF, set your default directory to SYS\$MANAGER and invoke the command procedure SYS\$MANAGER:ALFMAINT.COM, as demonstrated below:

```
$ SET DEFAULT SYS$MANAGER
$ @ALFMAINT
Terminal?
```

The ALF is an indexed file. It is updated in place and does not need to be purged. Backing it up regularly after a modification is a good practice.

In the next example the user JONES is associated with the terminal TTA1:

```
$ @ALFMAINT
Terminal (ddcu)? TTA1          ! Assigned terminal
Username? JONES
Terminal (ddcu)? EXIT
```

In the next example, a series of terminals are all set up for automatic login for a turnkey application:

```
$ @ALFMAINT
Terminal (ddcu)? TTA0           ! All terminals
Username? INVENTORY            ! on automatic
Terminal (ddcu)? TTA1           ! login
Username? INVENTORY
Terminal (ddcu)? TTA2
Username? INVENTORY
Terminal (ddcu)? TTA3
Username? INVENTORY
Terminal (ddcu)? EXIT
```

### 5.2.9.5 Logging in on an Automatic Login Terminal

Once you set up a terminal for automatic login, it can only be used for the designated account. This is most useful for applications terminals planned for use by persons who need not be very familiar with computers. Thus, an automatic login account will very likely also be a captive account.

The effect of the automatic login feature is to suppress the prompt for username. All other login features (system password, primary and secondary passwords, messages, and so forth) function normally, if enabled. You might consider disabling some of the messages as a service to your less "computer literate" users.

Passwords are optional, depending on the situation. If you want the account to be open to all users where the terminals are located, you can eliminate the password. When no password is required, the user has no data to enter at login. VAX/VMS logs the terminal in automatically in response to the BREAK key or the RETURN key and immediately enters the application, if the account is captive.

### 5.2.9.6 Protecting Automatic Login Accounts

Automatic login accounts are potentially accessible from terminals and sources other than the terminals listed in the ALF, and, therefore, require protection, especially if they have no password. Two steps you should take follow:

- 1 Restrict network and dialup access, as appropriate, with the AUTHORIZE qualifiers /NODIALUP, /NONETWORK, and /NOREMOTE.

- 2 Set the AUTOLOGIN flag in the account's UAF record. This flag makes the account available only by autologin, batch, and network proxy.

---

## 5.3 Authorizing Usage

As you authorize users, you must consider what restrictions should apply to each to provide additional control over the uses that they can make of the system. Typically you decide whether to restrict them to certain devices, certain commands, certain privileges, short account durations, certain working times, and/or certain modes of operation (batch, dialup, remote, network, local, or interactive).

---

### 5.3.1 Restricting Devices

There are a number of ways you can restrict the devices at the user's disposal. You may want to limit use to particular devices, or you may want to limit the amount of usage. The next few sections describe the controls available to you for restricting the usage of terminals, disk volumes, applications terminals, and miscellaneous devices.

---

**5.3.1.1 Restricting Terminal Usage**

VAX/VMS normally sets up terminals to be accessible to the SYSTEM account only, through the SYSGEN parameters TTY\_DEFPROT and TTY\_OWNER. (Users can login because login always starts to execute as a system process.) To make terminals accessible to certain users as applications terminals, you may want to change any or all of the following:

- Protection
- Owner UIC
- ACLs

The application of system passwords limits the use of those terminals to users who know the system password. You can also incorporate SET PROTECTION/DEVICE and SET ACL /OBJECT=DEVICE commands for specific terminals (with appropriate protection codes) in the command procedure SYS\$MANAGER:SYSTARTUP.COM.

---

**5.3.1.2 Restricting Disk Volumes**

You identify the user's default device and directory in the UAF record with the AUTHORIZE qualifiers /DEVICE and /DIRECTORY. You can limit the number of blocks available to the user on that disk (and any other disk) through the disk quota feature. (You use the VAX/VMS Disk Quota Utility (DISKQUOTA) that is described in the *VAX/VMS Utilities Reference Volume*.)

The volume protection in place on other disks controls how much access this user can obtain to the disks. The user's privileges, which can be extended or limited through the AUTHORIZE qualifier /PRIVILEGES, also influence the access that is ultimately acquired. (See Section 5.3.6.)

---

**5.3.1.3 Applications Terminals and Miscellaneous Devices**

You can use the DCL command SET PROTECTION/DEVICE to limit the access to any non-file-structured device. You might also apply an access control list on the device to refine the limits on the user's access. Section 4.3.4.1 includes an illustration of these techniques.

### 5.3.2 Restricting Work Times

If you decide you cannot allow users to log in during certain periods of the day, you can set up the qualifiers that AUTHORIZE provides for this accordingly. You define primary and secondary days of the week with the /PRIMEDAYS qualifier, unless you conform to the default where primary days are Monday through Friday and secondary days are Saturday and Sunday. For example, if a user works Tuesday through Saturday, you would specify the /PRIMEDAYS qualifier as follows:

```
/PRIMEDAYS=(NOMON,TUES,WED,THUR,FRI,SAT,NOSUN)
```

Occasionally an operational change occurs that conflicts with the normal day assignments at your site, such as a holiday falling on a primary day. It is simple to override the normal day assignment if you have the OPER privilege. You use the DCL command SET DAY and specify the day type interpretation you desire for the current day. Note that this change applies to all logged in users, as well as those who will log in during the day. That is, users who are currently logged in who are unauthorized for logins for the day type once it changes, will be dumped off the system at the next hour. (The VAX/VMS Job Controller wakes up hourly and enforces the work time restrictions.)

Generally, you must first decide which types of login access should be restricted to certain hours, since you specify the work period with the qualifier that describes the login access. The login access qualifiers are: /LOCAL, /REMOTE, /DIALUP, /INTERACTIVE, /BATCH, and /NETWORK. However, if your site will apply one set of primary and secondary hours for all types of logins, your problem becomes much simpler. You just specify one qualifier, /ACCESS; the hours you include with this qualifier will apply to all modes of access.

Suppose you want to disable the user from running batch jobs during normal working hours. You would include the following qualifier when you define the user's account:

```
/NOBATCH=(PRIMARY, 9-17)
```

This specification permits the user to run batch jobs only during the hours of 6:00 p.m. through 8:59 a.m. on primary days, but all day long for secondary days.

You may be motivated to impose these types of restrictions if you feel users must be supervised and you lack adequate supervision during certain periods, or if you uncover evidence of damage occurring during a particular period. Other more innocuous reasons include wanting to balance the workload on your system better.

There is another technique that is sometimes useful for specific accounts that you want to ensure are only used at times you authorize. You disable the account with the qualifier `/FLAGS=DISUSER`. The user will be unable to login until you reinstate the account with the `/FLAGS=NODISUSER` qualifier. This, in fact, is the technique recommended in the *Guide to VAX/VMS System Management and Daily Operations* for the SYSTEST and FIELD accounts when they are not in use.

---

### 5.3.3 Restricting Mode of Operation

Sometimes it is very helpful to restrict users to certain modes of operation. For example, the following concerns might prompt you to prohibit access through the network for some of your users:

- The user has data that should only be accessed through the local node.
- Penetration attempts are more likely to occur over a network because of the increased anonymity of the connection. (This concern is equally relevant to dialup connections as well.)

Similarly, you may want to limit the submission of batch jobs. Whatever your reasons, you use the AUTHORIZE qualifiers pertaining to access modes to impose the restrictions.

For example, the security manager could prohibit user LWHITE from submitting batch jobs by simply including the `/NOBATCH` qualifier when opening or modifying the LWHITE account.

---

### 5.3.4 Restricting Command Usage

There are several methods that you can apply to affect the command usage by your users. Among them are the following:

- Build another copy of the DCL tables and remove or modify command definitions, as you see fit. (You use the VAX/VMS Command Definition Utility, which is described in the *VAX/VMS Utilities Reference Volume*.) Use the /CLITABLES qualifier in the user's UAF record to specify the modified tables. Specify /FLAGS=DEFCLI to ensure that the user can only log in with the specified CLI and tables.
- Impose ACLs on the system program files in the library SYS\$SYSROOT:[SYSEXEC].

---

### 5.3.5 Restricting Account Duration

It is good practice to set an expiration time for an account that matches the maximum length of time you expect the user to require the access. In this way, your purpose is served when the expiration time arrives, even if you are delayed in deleting the account for a few days; the system prohibits access to the account automatically for you. Despite this action, you must eventually remove the UAF record and delete the user's files. VAX/VMS only affects the user's ability to access the files once the account expiration time is reached; it does not delete the files or remove the user's UAF record.

To set the account expiration time, use the /EXPIRATION qualifier in the user's UAF record with AUTHORIZE. For example, the following qualifier specifies that the user's account will expire on the 30th of June, 1985:

```
/EXPIRATION=30-JUN-1985
```



### 5.3.6 Granting User Privileges

Privileges restrict the performance of certain system activities to certain users. These restrictions protect the integrity of the operating system's performance and thus the integrity of service provided to users. You should grant privileges to each user on the basis of two factors: (1) whether the user has a legitimate need for the privilege, and (2) whether the user has the skill and experience to use the privilege without disrupting the system. Never issue a privilege to a user who has no legitimate need for the privilege. You can always grant the privilege later, when the need arises and the justification is clearer. That way, you are also more likely to remember to revoke the privilege when the need has passed.

Privileges fall into seven categories according to the damage that the user possessing them could cause the system:

- None—No privileges
- Normal—Minimum privileges to effectively use the system
- Group—Potential to interfere with members of the same group
- Devour—Potential to consume noncritical system-wide resources
- System—Potential to interfere with normal system operation
- File—Potential to compromise file security
- All—Potential to control the system

A user cannot execute an image that requires a privilege that the user does not possess—unless the image is installed as a known image with the privilege in question. (See the *Guide to VAX/VMS System Management and Daily Operations* for instructions on installing known images.) Execution of a known image with privileges grants those privileges to the user process executing the image—on a temporary basis, for the duration of the image's execution. Thus, you should install user images with amplified privileges only after ensuring that the user needs the access and is unlikely to misuse it.

A user's privileges are recorded in the user's UAF record in two 64-bit privilege vectors. One vector stores the authorized privileges and the other vector stores the default privileges. These two masks become, respectively, the process' authorized and current privileges.

When a user logs in to the system, the user's privilege vector is stored in the header of the user's process. In this way, the user's privileges are passed on to the process created for the user. Users can use the DCL command SET PROCESS /PRIVILEGES to enable and disable privileges for which they are authorized, to further control the privileges available to the images they run. Moreover, any user with the SETPRV privilege can enable any privilege.

Table 5-1 categorizes the privileges and includes a brief, general definition of the powers associated with each privilege.

**Table 5-1 VAX/VMS Privileges**

Category	Privilege	Activity Permitted
None	None	None requiring privileges
Normal	MOUNT	Execute mount volume QIO
	NETMBX	Create network connections
	TMPMBX	Create temporary mailbox
Group	GROUP	Control processes in the same group
	GRPPRV	Group access through SYSTEM protection field
Devour	ACNT	Disable accounting
	ALLSPOOL	Allocate spooled devices
	BUGCHK	Make bugcheck error log entries
	EXQUOTA	Exceed disk quotas
	GRPNAM	Insert group logical names in the name table
	PRMCEB	Create/delete permanent common event flag clusters
	PRMGBL	Create permanent global sections
	PRMMBX	Create permanent mailboxes
	SHMEM	Create/delete structures in shared memory
System	ALTPRI	Set base priority higher than allotment
	OPER	Perform operator functions
	PSWAPM	Change process swap mode
	WORLD	Control any process
	SECURITY	Perform security related functions
	SHARE	Access devices allocated to other users
	SYSLCK	Lock system-wide resources

**Table 5-1 (Cont.) VAX/VMS Privileges**

Category	Privilege	Activity Permitted
Files	DIAGNOSE	Diagnose devices
	SYSGBL	Create system-wide global sections
All	VOLPRO	Override volume protection
	BYPASS	Disregard protection
	CMEXEC	Change to executive mode
	CMKRNL	Change to kernel mode
	DETACH	Create detached processes of arbitrary UIC
	LOG_IO	Issue logical I/O requests
	PFNMAP	Map to specific physical pages
	PHY_IO	Issue physical I/O requests
	READALL	Possess read access to everything
	SETPRV	Enable any privilege
	SYSNAM	Insert system logical names in the name table
	SYSPRV	Access objects through SYSTEM protection field

**5.3.6.1****Alternatives to Privilege**

Privileges serve an important purpose by allowing or disallowing individual users to perform certain functions. However, security managers frequently make the mistake of overextending privileges to users because a few seldom-performed functions require the privileges and seem to justify granting them. The problem is that once you grant a privilege in the UAF, the user possesses it, whether actively using it or not. Users may not be aware that they can turn off or on any of their authorized privileges with the DCL command SET PROCESS/PRIVILEGES. You should advise users to disable all privileges as soon as they are not needed during a session.



ZK-2074-84

Once you overcome the problem of potential abuse of privileges by the users who rightfully hold them, you face concerns that outsiders may acquire them. Outsiders can acquire privileges illegally either by breaking into an account or through the introduction of programs or command procedures of the Trojan horse variety, that are designed to pass privileges on. By limiting the set of privileges that each user holds to the barest minimum, you can hope to contain the damage that outsiders can inflict if they do break in. Naturally, you should expect outsiders to target your most privileged accounts for their heaviest assaults. Consequently, it is wise to disguise these accounts a little if you can. That is, avoid usernames like OPERATIONS, SECURITY, RECORDS, MANAGEMENT, and so on, that are certain to provoke undesirable outside interest.

There are a few alternatives that might prove preferable to granting certain privileges. For example, certain users might need to run a particular program that requires one of the more powerful privileges, but that is their only justification for needing the privilege. You could install the program with the necessary privilege using the VAX/VMS Install Utility. However, you would then put an ACL on the executable image file to clearly specify only those users allowed to execute it. The users would effectively possess the privilege only during

the interim when they are actually executing the image. When the image stops running, the user no longer holds the privilege.

The following example should help clarify this technique. The program SDA.EXE (which is required to analyze system dumps) requires CMKRNL privilege to analyze the running system. The security manager installs SDA.EXE with the CMKRNL privilege, as follows:

```
$ RUN SYS$SYSTEM:INSTALL  
SDA.EXE /PRIVILEGED=CMKRNL  
{messages}
```

Next, the security manager places an ACL on SDA.EXE and also sets the UIC-based protection to deny all access to the WORLD category of users, as follows:

```
$ SET ACL/ACL=(IDENTIFIER=SDAUSERS,ACCESS=EXECUTE) SYS$SYSTEM:SDA.EXE  
$ SET PROTECTION=(WORLD) SYS$SYSTEM:SDA.EXE
```

Finally, the security manager uses the AUTHORIZE command SHOW/IDENTIFIER=SDAUSERS to confirm that the users who hold the SDAUSERS identifier are those intended to run the program. If necessary, the manager makes adjustments to this list of users with AUTHORIZE.

Although DIGITAL ensures that all system programs (such as SDA) that are supplied with VAX/VMS are linked with the /NOTRACE qualifier to prevent online debugging or traceback, do not overlook this important step with your own images.

**Note: All images that you install with privilege must be linked with the /NOTRACE qualifier to prevent online debugging and traceback.**

Both online debugging and traceback practically invite clever users to violate your system, which makes them extremely dangerous when available for images that run with privileges. All sorts of undesirable activities can be initiated by a user who succeeds in running a sufficiently privileged image with debugging or traceback enabled. After gaining control, the skillful user can modify the system or its files (such as the UAF records), substitute software, examine operating system internals, obtain additional privileges, and so forth. (Nor would you want to see what the semi-skilled user could do!) The opportunities are almost unlimited and the damage can be catastrophic.

Another alternative to the wholesale granting of privileges is to set up emergency or specialized privileged accounts. Users would only log in to these privileged accounts to perform certain very specific functions when required. There are two variations on this theme. In one case, there is a limited group who know about the account and are informed how to use it. In the other case, the security manager creates two accounts for the user, giving the privileges to one account but not to the other. In this case, the user would have the same UIC and the same default directory in each account. (This is the only case where DIGITAL recommends shared UICs, since there is still only one actual user.) If you decide to adopt this dual account practice, try to avoid obvious usernames that reveal which account is the privileged account. For example, you would be wiser to set up the general purpose account for Robert Wesley Dogwood as ROB\_DOGWOOD and the privileged account as WES\_DOGWOOD, than to select usernames of DOGWOOD and DOGWOOD\_PRIV.

In both cases, the manager could place special restrictions on the privileged account, such as long passwords, brief password lifetimes, restricted hours, and limited modes of operation (no dialups, no network, remote, or batch access). Such accounts might also be good candidates for limited account durations, to force frequent reconsideration of the privilege requirements.

### 5.3.6.2

#### **Suggested Privilege Allocations**

Appendix A provides reference material regarding all the user privileges, including recommendations on the appropriate reasons to grant them. Security managers should study the descriptions very carefully and understand the recommendations. When faced with allocating user privileges, managers should consult Table 5-1 and be conservative.

The summary guidelines in Table 5-2 might prove helpful, since they indicate the **minimum** privilege requirements for common classes of system users. If you adhere to these suggestions, you will not overextend privileges in most cases. However, there really is no substitute for spending time in careful consideration of individual privilege requirements based on actual system access needs.

**Table 5-2 Minimum Privileges For System Users**

Type of User	Minimum Privileges
General Users	TMBMBX,NETMBX
Operators	OPER
Group Managers	GROUP,GRPPRV
System Manager/Administrator	SYSPRV,SETPRV
Security Manager	SETPRV,SECURITY

**5.3.6.3 Controlling Privileged Accounts**

Since abuse of privileged accounts can result in such serious losses, security managers might impose special controls on accounts with the most powerful privileges. The following suggestions are worthy of consideration:

- Limit access to the account. For example, you can prohibit dialup or network access with the /NODIALUP or /NONETWORK qualifiers, respectively, to discourage outsiders from attempting breakins from remote locations where their activities might be less subject to detection and prosecution.
- Use the /PRIMEDAYS and /NOACCESS qualifiers to restrict the time of day or days of the week that logins can be performed into the account. Select periods of time that can be properly monitored for appropriate use.
- Disable the account when not in use with the AUTHORIZE qualifier /FLAGS=DISUSER.
- Use a captive login command procedure for additional validation.

Another important technique is to impose security alarms to detect abuses of the privileges pertaining to file protection: BYPASS, SYSPRV, READALL, and GRPPRV. In this application of alarms, which is very similar to the example in Section 4.9.2.2, you select key files that are likely targets for illegal access and set up security alarm ACEs in the ACLs of these key files. You use the DCL command SET AUDIT to specify the privilege usage to detect and report. You then observe all alarm messages that are delivered to the security operator terminals. As in the example, the use of a privilege



to gain access to the file is noted on a separate line. Whenever you find a privilege invoked, check the user's UAF record to confirm that the user legitimately has the privilege. If not, you should immediately suspect that the user has employed devious means such as a Trojan horse type of command procedure or program to acquire the privilege. Note, however, that less-sinister alarm messages also occur when VAX/VMS uses privileges to execute privileged images. An example would be the delivery of electronic mail with the VAX/VMS Mail Utility which requires the SYSPRV privilege.

---

### 5.3.6.4 Special-purpose Privileged Captive Accounts

Although in general you want to avoid granting privileges to captive accounts, there are situations where privileges must be granted and it may be better to grant them to tightly controlled accounts than to less restricted accounts. For example, users who perform backup operations will require the READALL privilege. This is a powerful privilege. By making the account that performs backups captive, you can ensure that the procedures are carried out according to your system's backup policy.

Refer to Section 5.8 for guidelines in setting up captive accounts.

---

### 5.3.7 Examples of Establishing User Accounts

This section illustrates the creation of three different user accounts. As you examine the options of these three accounts, you will observe they range from least restrictive to most restrictive. Chapter 7 includes a similar example that illustrates a number of principles involved in designing and implementing proxy login accounts.

The first account represents highly privileged users such as system managers, with minimum restrictions and maximum access to the system. However, you should note that although the owner of the account is certainly trusted, numerous aspects of password protection are imposed to protect such an attractive account from assaults by outsiders.

The second example illustrates an interactive user account with moderate restrictions, more likely typical of an account at a commercial site where security is a concern and the average user has limited access.

The third example depicts an applications production account, where the user is highly restricted.

In the examples below, you should assume that any value not specified defaults to the value provided by the DEFAULT record in the UAF, as shipped with VAX/VMS.

### 5.3.7.1

#### A System Manager's Account

The example in Figure 5-6 illustrates a number of AUTHORIZE qualifiers that would be appropriate for a system manager's account. Notice the use of a short password lifetime, the requirement that the automatic password generator be used to change passwords, and the use of primary and secondary passwords. These measures are important to protect the account because it affords many valuable privileges and access rights.

**Figure 5-6 Example of a Security/System Manager's Account**

```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF> ADD RIRONWOOD/PASSWORD=(VALTERSY,ESTREMA)/UIC=[001,100] -
_/DEVICE=SYS$SYSDEVICE/DIRECTORY=[RIRONWOOD] -
_/OWNER="Russ Ironwood"/ACCOUNT=SECURITY -
_/FLAGS=(AUDIT,GENPWD) -
_/PWLIFETIME=30-/PWDMINIMUM=8 -
_/PRIVILEGES=SETPRV
identifier for value:[000001,000100] added to RIGHTSLIST.DAT
UAF>
```

This user has the most powerful privilege of all: SETPRV. With this privilege the owner can acquire any other privilege, as needed.

### 5.3.7.2 A Typical Interactive User's Account

Figure 5-7 illustrates the creation of an account that might represent a fairly typical interactive user. Only one password is required, the password does have a minimum length, but it has a fairly long lifetime as compared to the manager's password. The user is allowed access during the week and on Saturdays, during a fifteen-hour time period.

**Figure 5-7 Example of a Typical Interactive User Account**

```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF> ADD RDOGWOOD /PASSWORD=TRALAYAM/UIC=[231,010] -
_ /DEVICE=BOTANYDEV/DIRECTORY=[RDOGWOOD] -
_ /OWNER="Robert Dogwood"/ACCOUNT=BOTNYDPT -
_ /FLAGS=(GENPWD) -
_ /PRIMEDAYS=(MON,TUES,WED,THURS,FRI,SAT,NOSUN) -
_ /EXPIRATION=15-JUNE-1985/PWDLIFETIME=90-/PWDMINIMUM=6 -
_ /NOACCESS=(PRIMARY,23-6,SECONDARY)/NODIALUP
identifier for value:[000231,000010] added to RIGHTS.LIST.DAT
UAF>
```

### 5.3.7.3 A Production Account

Figure 5-8 illustrates the creation of a production account. This account is designed to perform just one function: to roll up the grades at Treetown State and to produce mailings to each student's home. This job can only be run from the captive account REPGRADES, and it may not be run over dialup lines or as a remote job. Nor will the account permit network access. When the job is run through a local login, it is restricted to the hours of 8 a.m. through 5:59 p.m., Monday through Friday. However, when the job is run in the batch mode, it is not restricted to special times. The user who initiates the login must specify the password, GROBWACH. (Most likely only the security manager will change the password.) The process runs under the control of a special login command procedure (GRADES.COM), which presumably provides the operator with a menu of functions. (The process is restricted to the commands defined in the CLI table, GRADES\_TABLES.)

---

**Figure 5-8 Example of a Production Account**


---

```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF> ADD REPGRADES /PASSWORD="GROBWACH"/UIC=[777,031] -
_/DEVICE=ADMINDEV/DIRECTORY=[REPGRADES] -
_/OWNER="Campus Admin"/ACCOUNT=ADMIN -
_/FLAGS=(CAPTIVE,DISWELCOME,DISNEWMAIL,DISMAIL,DEFCLI) -
_/LOCAL=(PRIMARY, 8-17)/PRIMEDAYS=(MON,TUES,WED,THU,FRI,NOSAT,NOSUN) -
_/NONETWORK/NOREMOTE/NODIALUP -
_/LGICMD=GRADES/CLITABLES=GRADES_TABLES -
-
```

```
user record successfully added
identifier for value:[000777,000031] added to RIGHTSLLIST.DAT
```

---

### 5.3.8 Educating the New User

You will find that time spent educating new users will bring unexpected dividends in security. Your goal should be to make your expectations for proper usage, including applicable restrictions, as clear as possible, so that compliance will be voluntary and reliable. When users really understand what is expected of them, how to help, and why certain steps are helpful, the vast majority are very willing to help and to watch for indications that something is awry.

Education is your biggest ally in controlling the problem of user irresponsibility that Chapter 1 describes. If you fail to educate your users, you can bring on several additional problems you might not immediately foresee. For example, users can always cite ignorance as an excuse for problem behaviors, and you have little immediate recourse. You may find yourself extending second and even third chances while putting your system at great risk, until you are in a solid position to take the severe corrective disciplinary action called for in clear cases of user irresponsibility, probing, or penetration. However, cases of malicious behavior are more promptly detected and resolved when you have given the user ample directions, warnings, and help—right from the very beginning.

Each new user will need certain common pieces of information. To be certain you always provide all the information, you may want to rely on a standard form, such as the sample one in Figure 5-9. However, you should not simply hand the user a form or two and consider the job done. Rather, plan to discuss each point and spend time in live demonstrations on the system, to make the information clear.

No matter how you choose to present the information, the key topics you should cover include the following:

- 1 What is the location of the account? Specifically, which system, where is it located, what is the proper node name if on a network, and if the system is part of a cluster, what other nodes are available?
- 2 Which terminals may be used for logging in and where are they located?
- 3 Is the account restricted with regard to local, dialup, remote, interactive, network, or batch operations? If so, describe both the permitted use and the restrictions.
- 4 Can the account be accessed by dialing in? If so, provide the access telephone number and describe the procedure. Specify how many retries will be allowed and the maximum number of seconds allowed between each before the connection will be lost.
- 5 Are system passwords implemented for any terminals that the user may be using? If so, be sure to describe which terminals, how often the system password is changed, and how the user can learn the new system password.
- 6 What is the account duration; that is, will the account expire after a certain period of time elapses, such as a semester? Who should be seen to request an extension?
- 7 What is the username? What identifiers are held by the user, if any? What are the group and member numbers associated with the user?
- 8 What basic user password information is required? Specifically, what is the initial password? Is the password locked? If the password is not locked, instruct the user to change it immediately after logging in and direct how soon the password must be changed again. What is the minimum

length for the password? Is there a secondary password for this account and who will know it? Is the user free to select passwords or must they be automatically generated? (Take care not to divulge any passwords by writing them on the form, however.)

- 9** What is the default device and directory?
- 10** What is the default protection?
- 11** Are there quotas on the disk usage? If so, what are the values?
- 12** Are there restrictions on use? For example, are there certain days or hours of the day that are suggested or enforced? Explain primary and secondary days if applicable.
- 13** Are there files or directories that are shared? If so, provide the details.
- 14** Are there ACLs in force that affect the user? What identifiers does the user need to know?
- 15** Which privileges does the user hold?
- 16** What is the command language interface?
- 17** Which type of account is this—open, locked, captive, or normal?
- 18** Which nodes permit proxy logins for this user, if any?
- 19** What are the names of the queues the user may need to use?
- 20** Summarize the rules to observe. You should also describe in this section any behavior you would like observed by all users that you have not covered previously. For example, here you might describe actions related to the physical aspects of security, such as locking up materials, and so forth.

## **Security for the System Manager**

Figure 5-9 is an example of a form issued to a new botany professor joining the staff of Treetown State University. It covers the items indicated in the checklist above and would help prompt meaningful discussions between the professor and the security manager, about system security and the use of the new account.

---

**Figure 5-9 Sample Form Introducing a New Account**

---

TREETOWN STATE UNIVERSITY  
Department of Computer Operations

Dear Professor Dogwood

-----  
Welcome to the staff of the Horticultural Department at Treetown State University. You will be using our VAX/VMS systems for class work, grading, preparing personnel reports, composing and scoring tests, as well as other personal uses of your own choosing.

An account has been created for you under the user name of RDOGWOOD. You will be able to access any of the three nodes on the ARBORETUM cluster: WILLOW, MAPLE, and ELM. You can login from any of the terminals located in the laboratory areas known as the Greenhouse and the Hothouse. You will also be logging in from the terminal in your office, which is TTE7. Dialups from remote locations are not currently authorized. Your account expires at the end of the academic year, but will be renewed when your new contract is signed and returned to Dr. Charteroak's office.

All terminals are secured with a system password, which changes the first Monday of every month or when deemed necessary. (Change notices will be posted on the laboratory doors. You can obtain the system password from Mary Beth in Dr. Charteroak's office by presenting your university I. D. card.)

You must also enter a user password to login. This password must be a minimum of 6 characters long. A series of password choices will be automatically generated for you when you issue the SET PASSWORD command, and you will be restricted to passwords suggested by the system. Your password will expire every 90 days. We will log in together for the first time, at which time I will provide your initial password---which you must immediately change. (For your convenience, the login procedures that we will use are repeated on the attachment.) You will be responsible for the security of your password, and you are requested not to share it or post it where it might be discovered.

---

(Continued on next page)



---

**Figure 5-9 (Cont.) Sample Form Introducing a New Account**

---

Your User Identification Code (UIC) is [BOTSTAFF,RDOGWOOD].  
You hold the following identifiers: RDOGWOOD, GRADES, BIDEPT,  
TESTS, and UNIVRSTY. You can ensure restricted access to your  
files for specific staff members by using their usernames as  
identifiers in your access control lists.

Your default device is BOTANYDEV and your default directory is  
RDOGWOOD. You are restricted to 5000 blocks on BOTANYDEV.  
By default, your files will be created with the following  
standard protection: (S:RWED,O:RWED,G:RE,W).

You have unrestricted access to the cluster 15-hours a day,  
6 days a week. That is, you may login to the system between  
the hours of 7 a.m. and 10 p.m., Monday through Saturday.

You hold the following privileges: GROUP, TMPMBX, and NETMBX.

You will use DCL as your command language.

You are allowed proxy logins on node LILAC, as user BIDEPT.  
You can log in to any of the nodes on the cluster (WILLOW, MAPLE  
and ELM) as user RDOGWOOD. You can acquire network access to  
the node ASH by requesting it through Dr. Charteroak's office.

In addition to keeping your password secret, please keep this  
material locked up in a file in your office. Always log out  
before you walk away from any terminal. Lock your office  
when you leave for more than 15 minutes. Advise Mary Beth if  
any graduate students will be allowed to use your terminal  
and when.

Thank you for your cooperation with these measures, which will  
ensure secure yet reasonably open access for us all.

You can reach me with any questions on extension 2287 or  
as user RIRONWOOD on the cluster. You should contact your  
cluster manager, Rose Thorne (RTHORNE), in my absence.

Yours truly,  
*Russ T. Ironwood*  
Russ T. Ironwood *R.T.I.*  
Security Manager

## 5.4 Protecting Information

Once you have attained some confidence that you have a workable scheme whereby only desirable users can gain access to the system commands, you must address just which files you want your users to access. On every system there are files that require protection. Even on the most open system you will want protection for the system software, although DIGITAL provides default protection for this purpose that is generally sufficient.

Your tools for protecting information include the DCL commands SET PROTECTION, SET UIC, SET OWNER, SET DIRECTORY, SET FILE, EDIT/ACL, SET ACL, and SHOW ACL. These commands are described in the *VAX/VMS DCL Dictionary*. This section will discuss appropriate uses for many of these commands and the qualifiers you will apply most often.

Now return to the computer operations scene at Rainbow Paint. Suppose there is a personnel data file that is owned by Olive Green in the Executive group. (Olive's UIC is [EXEC,OGREEN].) Contrary to good file naming practice, this file has the very enticing name of PERSONNELDATA.DAT. The DCL command DIRECTORY/FULL in the example below reveals there are already two forms of protection for this file, standard UIC-based protection and an ACL.

```
$ DIRECTORY/FULL PERSONNELDATA.DAT

Directory RAINBWEEXEC:[OGREEN]
PERSONNELDATA.DAT;1      File ID: (10839,31,0)
Size: 321/322            Owner: [EXEC,OGREEN]
Created: 01-NOV-1984 16:42 Revised: 24-DEC-1984 17:29 (6)
Expires: 01-NOV-1985      Backup: 23-DEC-1984 20:35
File organization: Sequential
File attributes: Allocation: 350, Extend: 0, Global buffer count: 0, No version limit
Record format: Variable length, maximum 80 bytes
Record attributes: Carriage return carriage control
File protection: System:RWED, Owner:RWED, Group:, World:
Access Cntrl List: (IDENTIFIER=[EXEC,OGREEN],ACCESS=READ+WRITE+EXECUTE+DELETE+CONTROL)
                  (IDENTIFIER=[EXEC,SGOLD],ACCESS=READ)

Total of 1 file, 321/322 blocks.
```

This listing reveals that the ACL for PERSONNELDATA.DAT is designed to give READ, WRITE, EXECUTE, DELETE, and CONTROL access to Olive Green and READ only access to Sunny Gold. All other users are given no access.

Rainbow Paint merges with Old Paint and adds 30 new employees. Suddenly Olive needs a full personnel department. She promotes three of the new employees to her staff and opens up a new group, PERSNL. She remains in the EXEC group, but authorizes the new employees as members of the PERSNL group. She also creates the general identifier PERSONNEL, using the ADD/IDENTIFIER qualifier of the VAX/VMS Authorize Utility. She makes herself, Sandy Sienna, and Autumn Brauns holders of the PERSONNEL identifier with the GRANT/IDENTIFIER command. (The third employee, Dusty Rose, receives only the UIC-based identifier DROSE.)

Olive invokes the VAX/VMS ACL Editor and modifies the ACL for PERSONNELDATA.DAT to the following:

```
(IDENTIFIER=PERSONNEL, ACCESS=READ+WRITE+EXECUTE+DELETE+CONTROL)
(IDENTIFIER=SGOLD, ACCESS=READ)
```

Thus, user DROSE who is in the personnel group but holds no special identifiers could not access this file. User SGOLD who is in the executive group could access this file for reading only.

After a couple of months, things quiet down. Some of the employees leave. Olive decides that Autumn Brauns no longer needs WRITE access to PERSONNELDATA.DAT. While you might initially wonder if Olive's options are to create a new identifier, change the ACL, change Autumn's group, or change the standard UIC-based protection on the file, you quickly realize that it only makes sense in this case to remove Autumn as a holder of the PERSONNEL identifier. Autumn is going to stay with the company and she will continue to function in the PERSNL group. Olive still needs the protection the ACL provides and the access it grants to the other two holders of the PERSONNEL identifier. Olive easily accomplishes the change with the AUTHORIZE command REVOKE/IDENTIFIER, as follows:

```
UAF> REVOKE/IDENTIFIER ABRAUNS PERSONNEL
```

---

#### **5.4.1 Restricting Command Outputs**

You should be aware that some of the DCL commands behave in differing ways depending on the privileges that the user holds.

For example, unless a user holds the WORLD privilege, the SHOW USERS command limits the display to users in the requesting user's group. If the user holds the WORLD privilege, however, the display can include all the users currently on the system.

---

#### **5.4.2 Protecting System Programs and Databases**

Normally, DIGITAL delivers system programs and databases with adequate protection. However, if for any reason you are dissatisfied with the default protection, you can change it with the techniques outlined in Chapter 4, provided you have the necessary SYSPRV privilege. You might also add an ACL to any file that you decide needs additional protection. Appendix C presents the recommended protection codes for all system files. Your VAX/VMS software should have this set of protection codes following a correct installation. Examine your system files from time-to-time to ensure that this protection is maintained.

As indicated, DIGITAL provides default protection for the system programs that it provides. However, if you have a special requirement, you might examine the potential of ACLs for your needs. For example, you could consider using ACLs to restrict the use of system programs such as compilers. (Any number of considerations might prompt this action, ranging from performance to licensing issues.)

Another possible question worth investigating is: are there cases where you do not want some or all of your users able to initialize media? If so, you could put an ACL to good use on the system program SYS\$SYSTEM:INIT.EXE. You would ensure that you grant no access to the WORLD category in the UIC-based protection field. Then create an ACL for the file that grants access to the specific users you want.

Similarly, if a department in your company has paid for a license to a software product, you may want to make that software available to them, but not to others. You would ensure that the WORLD category receives no access through the standard UIC-based protection code and create an ACE in the ACL for that file that allows the access through the department's identifier.

You may also find that ACL protection is relevant to protect your applications databases, limiting the access to certain users.

---

### 5.4.3 Precautions to Take when Installing New Software

When you install new software, you must address several security concerns. You want to ensure that you are not admitting software that will in any way corrupt or undermine your usual security precautions. You must also consider whether or not to install the software with any privileges. When you install privileged software, you allow users to execute it whether or not they personally possess the required privilege. In effect, you extend the privilege to the process while it runs the software. While this offers some advantages with clear security implications, it also introduces several security-related dangers that you must confront. This section discusses security aspects of installing new software.

---

#### 5.4.3.1 Avoiding Trojan Horse Programs

There is a risk when you install any new piece of software that buried under an innocent shell is a very dangerous program. When the program is run by a user, such as an operator, who happens to have some highly desirable privileges, the program can perform any number of undesirable functions while it executes. For example, such programs are notorious devices for the following:

- Passing the privileges of the person who runs the program back to the author of the program
- Authorizing undesirable users
- Changing the protection on system files
- Patching the system

As mentioned earlier, this sort of program is known as a Trojan horse, since it parallels the old Greek legend. You perceive it to be one thing and let it into your system, only to discover too late, that it was designed to undermine your system security.

You really have only one line of defense to protect yourself from such a plight: make certain that all software that you install or that your user's use is from reputable sources. Feel free to reject suspicious looking software of unknown origin. Educate your users to this possibility so that they, too, will think twice when someone offers them a program or command procedure that seems attractive.



ZK-2069-84

### 5.4.3.2

#### Installing Programs with Privilege

Some software will require privilege to run. You have the choice of extending the privilege to all users you expect will need to run the software, or of installing the program with the required privileges. Section 5.3.6 describes these options in greater detail.

#### 5.4.3.3 Protecting Programs and Directories

In the security world, the *worm* is a cousin of the Trojan horse. Both scourges focus their attacks on command procedures and executable programs. However, there is one distinction. While the Trojan horse requires the active involvement of the recipient since the "gift horse" must be accepted, the worm can inflict itself on its victim without any cooperation. This presents both advantages and disadvantages to security managers.

A worm capitalizes on faulty file protection to work its way through the system. It modifies successive user's command procedures or executable programs. Then, when the user runs the command procedure or program, the modification (which is just another copy of the worm) propagates itself further through the system by making use of that user's access rights and privileges.

A favorite target for worm attacks is a user's login command procedure—for two reasons:

- Login command procedures are generally easy-to-modify DCL command procedures.
- Login command procedures are guaranteed to be executed frequently by the intended victims.

The power and complexity of the VAX/VMS ACL mechanism unfortunately presents a great opportunity for worms. Sloppy (or ill-conceived) patterns of file protection and sharing can create so-called "worm-holes" in the system, in which user A has write access to files belonging to user B, user B has access to user C, and so forth. This is just the fertile ground the worm seeks. A worm propagates itself along these lines of access (assuming they lead to command files or programs executed by their owners) over a far greater area than you would expect from just examining any individual user's access rights. Should a worm-hole lead to a privileged account, it provides the opportunity to penetrate the system from a relatively innocuous-looking starting point.

The victim of a worm attack may be totally unaware that the attack occurred (especially if the worm takes care to clean up after itself) and the user believes that all the normal precautions against Trojan horses have been taken. On the other hand, careful use of file protection is sufficient protection against worm attacks. The best policy is to ensure that likely targets are not modifiable by other users, regardless of whether you trust

them or not. For example, you should set up file protection so that your login command procedure permits no other access but READ access to all users except yourself. At the same time, make certain that the directory containing the login command procedure permits WRITE access only to users in the SYSTEM and OWNER categories.

While worms can possibly spread themselves throughout a system, propagating themselves as they go, they really attain their full potential for damage when they reach a target account with privileges. Therefore, users with privileges must be especially careful in protecting their files.

---

### 5.5 Disk Maintenance Considerations

Proper disk maintenance must consider the following topics:

- Physical security for disks
- Backups of disks
- Physical security for the backups
- How to retrieve files from backups

Questions of physical security, while extremely important to overall security, are not the domain of this guide. However, a few words about backups are appropriate.

The most important step any security manager can take to facilitate recoveries when files are deleted is putting an intelligent and carefully thought out set of backup procedures in place—prior to the event. You should study the VAX/VMS Backup Utility in the *VAX/VMS Utilities Reference Volume* for guidance in this. Proper application of the backup features prior to any mishaps will considerably improve your recoveries if accidental or malicious damage does occur.

When and if the time comes that you must retrieve files from your backup media, remember this cardinal security rule: **never** give your backup media to a user. The user could make a copy and then leisurely analyze and interpret the contents. Giving the media to a user could undermine much of your efforts to provide physical security for the media.



---

## 5.6 Methods for Discouraging Disk Scavenging

Disk scavenging is a concern of many sites whose security requirements fall in the medium to high range. It refers to the possibility an insider or outsider may read valuable data from a disk. While the types of protection described earlier go a long way toward preventing the reading of controlled files, consider the case of a disk that still holds magnetic imprints of data after the file header has been deleted in response to a purge or delete operation. The area may be overwritten at a later date, but in the meantime it is fallow ground for anyone who takes the time and effort to generate and use software that simply reads the data outside of VAX RMS control. In fact, sometimes disks are stolen from premises so that they can be read on highly sensitive devices that can glean information from faint magnetic impressions that persist even when an area is overwritten.

Combating the disk-scavenging problem requires a multipronged approach. You must do everything you can to make the environment secure. You also restrict access to disks that contain particularly valuable information, using the UIC-based volume protection. However, this technique merely slows down outsiders as well as the insider who tries to access the device over normal channels. Disk scavenging is more commonly done by authorized insiders or insiders who succeed in gaining those powers illegally. Thus, you also need to consider two features that VAX/VMS provides specifically to help thwart disk scavenging: data erasing and highwater marking.

---

### 5.6.1 Erasing Techniques

Files on the disk can be erased when they are deleted or purged. The inclusion of the /ERASE qualifier on the DELETE or PURGE commands will cause the system to write an erasure pattern of zeros once over the entire file location when a deletion or purge of that file occurs, respectively.

Security managers can take steps to encourage their users to specify their DCL commands as DELETE/ERASE and PURGE/ERASE. One method is to request voluntary compliance. If you include the following command definitions in the file with the logical name SYS\$SYLOGIN (normally SYS\$MANAGER:SYLOGIN.COM), you provide a little more impetus:

```
$ DEL*ETE :== "DELETE/ERASE"  
$ PUR*GE :== "PURGE/ERASE"
```

The determined user can bypass these definitions, of course, by adding the /NOERASE qualifier to either a DELETE or PURGE command, so this technique is really only helpful for forgetful users.

The more reliable technique of enforcing *erase-on-delete* (as this technique is sometimes known), is to turn on the feature for the entire volume. You use the DCL command SET VOLUME /ERASE\_ON\_DELETE for this purpose. When you take this step, **all** files on the volume are erased when they are deleted. (Before you go too far implementing this technique, you might investigate to determine whether erasures are really only important on certain disks. If so, you should apply the method judiciously.)

The DCL command INITIALIZE/ERASE serves two purposes. First, it completely erases the volume, thus making it safe to remove from the facility, recycle, or whatever. Second, it enables the erase-on-delete characteristic for the volume at volume initialization time.

VAX/VMS provides a default *data security erase (DSE)* pattern of zeros, that is applied during a single WRITE operation over the area. If you feel that the default pattern of zeros or the single rather than multiple number of erasures does not suit your requirements, you can modify either. Generally, for sites with high-level security requirements, a random pattern is better than a fixed pattern. Furthermore, the technology is already available that can detect and use faint residual magnetic impressions. Thus, if you conclude there is sufficient danger that a disk might be removed and read by some of this specialized analysis equipment, you might find rewriting the erasure pattern several times assumes special importance to you. You can learn how to customize the data security erase pattern to fit your needs by studying the information provided in the file SYS\$EXAMPLES:DOD\_ERAPAT.MAR.

## 5.6.2 Prevention Through Highwater Marking

The other method of fighting disk scavengers is called highwater marking. Section 4.7 introduces the concept of highwater marking. You may recall that the term highwater marking refers to a technique that attempts to track the furthest extent where each file has been written and to prohibit user attempts at reaching beyond that point.

The VAX/VMS implementation of this feature uses the principle of *erase-on-allocate* to achieve a similar end-result by a slightly different approach. That is, when a file is about to be created or extended, VAX/VMS determines how much disk space (the extent of the file) is required and applies the security erasure pattern to the areas (extents) it allocates for writing. The file is then written into the area that has just been erased for it. In this way, if any user subsequently gains access to the file (including its full extent), and attempts to read the area beyond where the file has been written to-date, all that is readable is the data security erase pattern. The file is protected to its highwater mark.

From the earlier discussion you learned that VAX/VMS turns highwater marking on by default for all volumes. The primary advantage of highwater marking is its effect as a deterrent to disk scavenging attempts. However, it has the disadvantage of requiring additional I/O, which will affect performance somewhat.

The greatest performance change attributable to highwater marking occurs when a disk is highly fragmented. Each separate extent that is erased requires an I/O operation. However, operating with highly fragmented disks is not in general desirable. (See the *Guide to VAX/VMS System Management and Daily Operations* and the description of the VAX/VMS Backup Utility in the *VAX/VMS Utilities Reference Volume* for suggestions regarding fragmented disks.)

You can turn off highwater marking on a volume-by-volume basis if you specify the DCL command SET VOLUME /NOHIGHWATER.

---

### 5.6.3 Summary of Prevention Techniques

Security managers can apply the following controls to discourage disk scavengers:

- Provide tight physical security, particularly around those disks with the most valuable information.
- Provide tight volume protection through UIC-based protection.
- Encourage the use of the /ERASE qualifier when key files are purged or deleted through user participation or volume enforcement.
- Permit the default highwater marking to occur on your most valuable disks.

While you gain a sense of security knowing that you have reduced the possibility of disk scavenging on your premises, you must be prepared to pay the price. Erasures are time consuming. Multiple erasures simply compound the effect. By judiciously applying the erasure requirements only on key designated disks or specific users, you may be able to achieve your goals without wreaking too much havoc on system performance.

Disk scavenging may still be possible when disks are removed from your site, depending on the technology applied to the problem. For sites with maximum security concerns, environmental security assumes even greater importance. It may be quite appropriate at such sites to require that all disks remain on the premises and that old disks be literally shredded prior to discarding.

---

## 5.7 Restricted Environments

There are a number of reasons why you may want to restrict the environment for an account. You may want semi-skilled users to perform routine tasks under tight computer control so that there will be limited opportunities for command errors or gaining additional access. You may want batch operations to run during hours where there will be little supervision. You may have applications such as printing payroll checks or reporting student grades that must be protected from outside intrusion. All of these situations might describe candidates for the establishment of captive accounts, sometimes referred to as

ted accounts or turnkey accounts. The main characteristic of such accounts is limited access to the system, generally through a specialized login command procedure. One example of a captive account appears in Section 5.3.7.3. However, this section will describe captive accounts in more detail, including two subcategories of captive accounts, guest accounts and proxy login accounts.

The term captive is altogether too reassuring. Captive accounts are actually frequent targets for successful assaults. It takes special skill to set up captive accounts that will thwart penetrators. In many cases, captive accounts run with command procedures that were written by someone other than the security manager. This presents a special problem, because poor design of the command procedures is a major cause of breakins. This section discusses programming techniques for the command procedures as if security managers were writing them. If you are not the writer, you should at least review the code, and you should reject any command procedure that fails to follow the recommendations found in this section.

---

### 5.7.1 Creating a Captive Account

You first define a captive account with `AUTHORIZE`, by including the following qualifier:

```
/FLAGS=(CAPTIVE)
```

This flag ensures that the account is noted as captive and disables `CTRL/Y` interrupts and prohibits user specification of the `/CLI` qualifier at login. Depending on other aspects of the application, you may also want to disable the welcome announcement and electronic mail, with three additional flags: `DISWELCOME`, `DISMAIL`, and `DISNEWMAIL`. Since the VAX/VMS Mail Utility has the ability to spawn other processes, it may behoove you to prohibit its use. However, the more general way to prevent the captive account user from spawning processes is to set the `AUTHORIZE` qualifier `/PRCLM` to 0.

You must decide whether or not users should be able to change the password for the account. You have two special password options that are appropriate with captive accounts:

- Dispense with passwords by setting the password null

- Lock the password with the `/FLAGS=LOCKPWD` qualifier so that only the security manager can change it

If you assign a locked password, you must reveal that password to all the users of the captive account. From a security standpoint, locked passwords are generally preferable to open accounts or accounts that let the user change the password. However, open accounts are sometimes appropriate for unsophisticated users or special applications. Allowing the users to change the password is also justifiable for other kinds of applications.

Your application may require you to impose additional `AUTHORIZE` qualifiers on the account, such as `/NODIALUP` to restrict the modes of operation. Also consider imposing restrictions for the periods of the day and days of the week when the process can run.

You might define a special set of DCL tables using the `/CLITABLES` qualifier. While using a restricted set of DCL tables will make it difficult for a user to exceed the intended limitations of the environment, this technique is not really secure. The sophisticated user may invoke the foreign command feature of DCL, which can only be suppressed by unsupported patches to DCL. It is, however, more efficient to define DCL tables than to resort to a DCL command procedure to emulate DCL. See the description of the `VAX/VMS Command Definition Utility` in the *VAX/VMS Utilities Reference Volume* for help in defining the DCL tables.

You should always give careful consideration to the question of privileges. Only in rare instances should you grant any privilege other than `TMPMBX` to a captive account.

Limit the disk quota for the captive account to the amount needed.

#### 5.7.1.1 Login Command File Considerations

The heart of the captive account is its login command procedure. You identify this file with the AUTHORIZE qualifier /LGICMD. Try to make certain that the login command file is tamperproof. If the account is allowed to either create files or perform other operations on them, make certain that neither the login command file nor its directory permits WRITE access to the UIC of this account. However, you must allow the UIC of this account READ access to the login command file. Also check that the guest account is not a member of the SYSTEM group, which is usually distinguishable by a group value less than 10 octal. However, be sure to check the SYSGEN parameter MAXSYSGROUP on your system for any difference.

To find out if the group UIC is unique, you can use the following form of the AUTHORIZE command SHOW:

```
SHOW [groupuic,*])
```

By keeping the guest account in a separate group, you can ensure that the account can only access its own and world-accessible files.

A useful technique with captive accounts is to have a related management account with the same UIC group, but a different UIC. This account can be well protected by frequent password changes, and controlled to only allow local logins. This account can then own the top-level directory and many of the files used by the captive account. The captive account can be allowed access to these files by controlling the file access through the GROUP category. If there is no requirement for the captive account to create files, there is no need for the captive account to have WRITE access to any directory, or to have any disk quota. If there is a requirement to allow file creation, you can give the captive account UIC an appropriate disk quota, as well as a subdirectory created with WRITE access for users in the GROUP category. It might prove convenient for the login command file for the account to set this subdirectory as the default.

One of your concerns is preventing the user from escaping from the command procedure. Since the main source of escape is through the DCL command CTRL/Y, the general strategy is to use the CAPTIVE flag to disable it for the account. In this way, you are assured that the user cannot use a CTRL/Y to break out of the captive account before the command procedure starts and possibly not until it finishes, either. Whether or not

the user ever obtains the ability to issue a CTRL/Y command, depends on the command procedure.

There are two types of captive applications where it is appropriate to allow the user to issue the CTRL/Y command subsequent to the startup of the command procedure. The first case concerns captive command procedures that want to provide their users with a CTRL/Y feature. These procedures would include appropriate ON CONTROL\_Y commands in the procedure. An example occurs in Figure 5-10. The second case concerns the captive command procedure that ultimately turns control over to the user. For example, consider a SYLOGIN.COM command procedure that performs additional security validation; its execution should be guaranteed to ensure its effectiveness. However, once SYLOGIN.COM has done its job, control can be turned over to the user. To achieve this sort of arrangement, you mark the account as captive and issue the DCL command SET CONTROL=Y when you are ready to release control to the user. Figure 5-11 provides an example.

If you intend to produce a highly controlled and restricted environment for the user, you must design the command procedure so that it runs in a loop until some exit condition occurs. Once the exit condition occurs, the command procedure logs the account out. Note that the command procedure must handle all possible error conditions, otherwise it might exit prematurely under some circumstances. Check the error condition handling carefully so that no error could cause it to loop indefinitely.

Do not allow the DCL command INQUIRE to appear in any of the command procedures. The INQUIRE command performs an evaluation while taking in input, which could create an opening to break the captive account. Instead use the DCL command READ/PROMPT. For example, to request the user to input the date, you might issue the following command:

```
READ/PROMPT="Enter date:" SYS$COMMAND DATE
```

For similar reasons you must avoid any use of the construction 'x, where x contains a string typed in by the user. Never permit a captive command procedure to attempt an evaluation of a symbol that the user enters. Clever applications of lexical functions could break the command procedure.



If the function of the command procedure requires text preparation, you may want to run an editor. If you do this, design the environment of the account with extra caution. Remember that most editors are capable of reading and writing arbitrary files (within the access rights of the account).

**Note:** Do not permit the captive account command procedure to use the TECO Editor; TECO has sufficient capabilities to break out of any captive command procedure.

Figure 5-10 presents an example of a command procedure that provides a completely captive environment, so that the user can be restricted to a very limited set of commands. Note that the security manager would use the AUTHORIZE qualifier /NOINTERACTIVE when establishing this account.

The sample captive command file in Figure 5-11 can be used on privileged accounts. The account should only be used interactively from a local terminal. Thus, the security manager would be certain to include the AUTHORIZE qualifiers /NODIALUP, /NOREMOTE, /NOBATCH, and /NONETWORK when establishing the account.

---

### 5.7.1.2

#### Guest Accounts

DIGITAL does not recommend the practice of setting up guest accounts. The usual justification for needing guest accounts is to permit users to send electronic mail or read reports. These needs can best be handled by special proxy login accounts, which should also be captive accounts. However, if you find you need a guest account for any purpose other than sending mail or reading reports over the network, you should take steps to make the guest account secure and captive. For more detail on each of the following steps, refer to the previous discussions of captive accounts.

- Use an obscure password for the guest account and change the password frequently. Never use easily guessable account name and passwords combinations such as GUEST/GUEST or USER/USER.
- Maintain a list of people who are allowed to use the account. (Changing the password regularly will help you to keep this list current.)
- Set up the guest account so that it is in a group by itself.

**Figure 5-10 Example of a Captive Command Procedure**

```

$ deassign sys$input
$ prev_sysinput == f$logical("SYS$INPUT")
$ on control_y then $goto next_cmd
$ set control=(y,t)
$next_cmd:
$ on error then $goto next_cmd
$ if prev_sysinput .nes. f$logical("SYS$INPUT") then deassign sys$input
$ read /end=next_cmd /prompt="$ " sys$command cmd
$!
$ delete = "delete"
$ delete /symbol /local /all
$ if f$locate ("@", cmd) .ne. f$length(cmd) then goto illegal_cmd
$ if f$locate ("=", cmd) .ne. f$length(cmd) then goto illegal_cmd
$ t1 = f$locate (" ", cmd)
$ cmd1 = f$extract (0, t1, cmd)
$ if f$locate (cmd1, "LOGOUT") .eq. 0 then goto logout
$ if f$locate (cmd1, "HELP") .eq. 0 then goto help
$!
$! Place other validation checks here
$!
$ write sys$output "%CAPTIVE-W-IVVERB, unrecognized command"
$ write sys$output "  \", cmd1, "\"
$ goto next_cmd
$!
$illegal_cmd:
$ write sys$output "%CAPTIVE-W-ILLEGAL, bad characters in command"
$ goto next_cmd
$!
$cmd_ok:
$ define sys$input sys$command:
$logout:
$ logout
$ goto next_cmd
$help:
$ help
$ goto next_cmd
$!
$! Place other prevalidated commands here
$!

```

- Also check that the guest account is not a member of the SYSTEM group.
- Place the default login command procedure in the directory SYS\$MANAGER. You can accomplish this with the AUTHORIZE command MODIFY, as follows:

```
MODIFY guest-account/LGICMD=SYS$MANAGER:filename.COM
```

**Figure 5-11 Sample Captive Procedure for Privileged Accounts**

```
$ if f$mode() .nes. "INTERACTIVE" then $logout
$ term = f$logical("SYS$COMMAND")
$ if f$locate("_T", term) .eq. 0 then $goto allow
$ if f$locate("_OP", term) .ne. 0 then $logout
$allow:
$ set control=(y,t)
```

- Make the account captive by setting the AUTHORIZE qualifier /FLAGS as follows:  
/FLAGS=(DISCTLY, LOCKPWD, CAPTIVE)
- Limit the number of subprocesses that the account can create to 0. Use the AUTHORIZE qualifier /PRCLM=0.
- The only privilege the guest account should receive is TMPMBX.
- Make certain that the default login command procedure has the following commands to handle error conditions:

```
$ SET ON
$ SET NOCONTROL
$ ON ERROR THEN LOGOUT/BRIEF
```

- If LOGOUT is defined as a global symbol and points to a command procedure (issue the DCL command SHOW SYMBOL LOGOUT to confirm this), include the following DCL command in the default login command procedure for the account:

```
DELETE/SYMBOL LOGOUT/GLOBAL
```

This will eliminate the possibility the user could break the captive account at logout time by typing **CTRL/Y**.

- Prevent outsiders from misusing your system resources through the submission of batch jobs under the guest account. You would include the AUTHORIZE qualifier /NOBATCH when you create the account.
- Limit the disk quota for the guest account UIC to the amount needed.
- Do not allow the DCL command INQUIRE to appear in any of the command procedures.

---

### 5.7.1.3 Proxy Login Accounts

Proxy login accounts should also generally be set up as captive accounts. Proxy login accounts permit remote users to access a local account without specifying a password. Section 7.6.1.2 describes proxy login accounts in detail. You will note that many of the recommendations conform to the advice presented here for captive accounts.

---

## 5.8 Auditing with Security Alarms

Security alarms are messages sent to the security operator's terminal when specific events take place. As the security manager, you can use alarms in your efforts to maintain a secure system. Alarms can help you detect outsiders' attempts to break into the system or monitor undesirable activity at your site. For example, you might enable an alarm that sends a message to the security operator's terminal whenever a UAF record changes.

In dealing with security alarms, your job as security manager requires that you carefully select and enable the events to be audited, that you enable a security operator terminal, and that you use the alarm information.

---

### 5.8.1 Alarm Events

The events that can be audited are:

- Selected types of access to files and global sections
- Event requested by an ACL on a file or global section
- Use of privilege to access files and global sections
- Installation of images
- Logins, logouts, and breakin attempts
- Modifications to the system and network UAF
- Changes to system and user passwords
- Modifications to the rights database
- Execution of the SET AUDIT command
- Volume mounts and dismounts

You select the events to be audited by specifying one or more keywords to the /ENABLE qualifier of the SET AUDIT command. See the *VAX/VMS DCL Dictionary* for more information.

If you enable alarms for all or many events, many alarm messages will be sent to the security terminal. The best practice is to implement alarms only for key events, just a few at a time. Enabling too many alarms results in the failure to monitor each alarm appropriately and fosters a lax attitude about alarms. While alarms can be a powerful tool when used judiciously, they quickly lose their attention-getting quality when overused.

See Section 6.2.2 for suggestions about which events you should enable if you suspect your system is under attack.

---

### 5.8.2 Enabling a Security Operator Terminal

Before you enable alarms for particular events, you should first enable a security operator's terminal. Choose a terminal that provides hard-copy output and is located in a secure location. The following DCL command enables the terminal from which the command is issued:

```
$REPLY/ENABLE=SECURITY
```

Any terminal may be enabled as a security operator; there may be one, none, or more than one such terminal. If you intend to make serious use of alarms, have them sent to a separate terminal and disable them on the system console. You accomplish this technique by adding the following to SYSTARTUP.COM:

```
$DEFINE/USER SYS$COMMAND OPA0:  
$REPLY/DISABLE=SECURITY  
$DEFINE/USER SYS$COMMAND TTA3:  
$REPLY/ENABLE=SECURITY
```

Note that regardless of whether any security operator terminals are enabled, security alarms still go into the operator log file.

---

### 5.8.3 Alarm Messages

After you enable a security operator terminal and enable specific alarm events with the SET AUDIT/ENABLE command, alarm messages are sent to the security operator terminal when the selected events occur. Security alarms have the format of the following sample alarm message:

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:27:52.26 XXXXXXXXXXXX ①  
Security alarm on LASSIE / System UAF record modification ②  
Time: 15-JUN-1985 12:27:52.25 ③  
PID: 23C00155 ④  
User Name: MENACE ④  
Rec Mod: GOWER  
Fields Mod: PRIVILEGES
```

The information included in the message depends on the type of event; however, in all cases, the alarm message contains these four elements:

- ① The OPCOM heading, which includes the date and time the alarm was sent
- ② The type of alarm event
- ③ The date and time the alarm event occurred
- ④ The perpetrator of the event, as identified by the username and process identification (PID)

Other information contained in the alarm messages is specific to the type of event that the alarm signaled. Appendix E includes examples of the alarm messages associated with particular alarm events.

---

### 5.8.4 Audit Reduction Facility

If you have enabled security alarms, the operating system writes information resulting from those alarms to the security operator's log file. Security alarms are effective only when you use that information; however, because you can enable alarms for many objects and types of access, the log file often contains a large volume of information. To selectively extract information from the operator's log file, you can use SECAUDIT.COM (a command procedure residing in SYS\$MANAGER).

To extract all of the security alarm information from the current operator's log file (SYS\$MANAGER:OPERATOR.LOG), execute the following command:

```
$ @SYS$MANAGER:SECAUDIT
```

Output from SECAUDIT is displayed on SYS\$OUTPUT. If you want to write the records to a file, you include the file specification with the /OUTPUT qualifier. The following command writes the records to the file BREAKINS.DAT in the user's current default directory:

```
$ @SYS$MANAGER:SECAUDIT/OUTPUT=BREAKINS.DAT
```

### 5.8.4.1

#### Optional Parameters

SECAUDIT.COM accepts five optional positional parameters:

- p1 Name of the log file that is scanned for the selected security alarm information. By default, SYS\$MANAGER:OPERATOR.LOG is searched.
- p2 Specific username for which the relevant security alarms are to be displayed.
- p3 Starting date and time of the first security alarm entry to be displayed.
- p4 Ending date and time of the last security alarm entry to be displayed.
- p5 Selection criteria - specify the selection criteria with the keywords used with the /ENABLE qualifier of the SET AUDIT command.

By default, SECAUDIT.COM searches SYS\$MANAGER:OPERATOR.LOG for security alarm information. Use the p1 parameter to specify a log file that is different from the default.

The remaining parameters select specific alarm information. Using these parameters, you can select alarm information generated in the following ways:

- By specific users
- Within a particular time frame

- By particular types of alarms

For example, the following command extracts all security alarm records generated by the user SARDONO after February 1, 1985.

```
$@SYS$MANAGER:SECAUDIT "" SARDONO 1-FEB-1985
```

Note that because the parameters are positional, you pass a null parameter by using a set of quotation marks as a place holder in the command string.

Use the p5 parameter to select alarm information that resulted from a particular type of event. The operating system audits a number of events, which are enabled by specifying keywords with the /ENABLE qualifier of the SET AUDIT command. Use the same keywords to select the alarm information from the operator's log.

For example, the following command extracts all security alarm records generated by breakin attempts, any access to a file using the SYSPRV privilege, or any access to a file using the BYPASS privilege.

```
$@SYS$MANAGER:SECAUDIT "" "" "" "" BREAKIN,FILE_ACCESS=(SYSPRV,BYPASS)
```

---

### 5.8.5 Auditing a Terminal Session

On occasion, you may need to audit an entire terminal session. If you set host to your own system and specify that a log file of the session be kept, the resulting log file will contain a record of the entire terminal session. For example, the following command keeps a record of the entire session in the log file APRIL15.LOG:

```
$SET HOST 0 /LOG=APRIL15.LOG
```

Note that if you use the /LOG qualifier without including a file specification, the log information is stored in the file SETHOST.LOG.

Use of the SET HOST command requires that DECnet be configured and started. The installation does not have to buy a DECnet license; the DECnet license is only necessary to actually communicate with remote nodes. To use SET HOST in an environment without DECnet, you must configure a network database with no communications lines or remote nodes, and



you must start the network. In the absence of supported communications equipment, NETCONFIG.COM will perform these steps correctly. Details on these operations are provided in the *Guide to Networking on VAX/VMS*.

### 5.8.6 Enforcing a Terminal Session Audit

You can enforce auditing of terminal sessions for selected users by use of a special captive account and appropriate command procedures. Users for whom session auditing is enforced must first log into the special captive account and then into their own account. The captive account assures that the session is audited. This section provides guidelines on how to set up the captive account (named USER\_AUDIT in this example) and includes samples of appropriate command procedures. The captive account USER\_AUDIT is set up as follows:

```
UAF> ADD USER_AUDIT /FLAGS=(CAPTIVE,DISMAIL,DISNEWMAIL) -
/LGICMD=SYS$SYSROOT:[USER_AUDIT]AUDITLOG -
/DEV=SYS$SYSROOT: /DIR=[USER_AUDIT] -
/NETWORK /NOBATCH /UIC=xxx ...
```

The AUDITLOG.COM command procedure enables auditing of the terminal session.

```
$ ! AUDITLOG.COM - log into specified account with terminal session
$ ! auditing enabled.
$ !
$ WRITE SYS$OUTPUT "Please log into the account of your choice."
$ WRITE SYS$OUTPUT "Your terminal session will be recorded."
$ WRITE SYS$OUTPUT ""
$ !
$ ! Acquire the intended username and save it in a temp file. Use it
$ ! to name the log file, and pass it as the first line of input to
$ ! LOGIN.
$ !
$ READ/PROMPT="Username: " SYS$COMMAND USERNAME
$ PID = F$GETJPI (0, "PID")
$ OPEN/WRITE OUTPUT USERNAME'PID'.TMP
$ WRITE OUTPUT USERNAME
$ CLOSE OUTPUT
$ DEFINE/USER SYS$INPUT USERNAME'PID'.TMP
$ SET HOST 0 /LOG='USERNAME'.LOG
$ DELETE USERNAME'PID'.TMP;0
$ LOGOUT
```

You must set up each account for which session auditing is to be enforced as follows:

```
UAF> MODIFY username /FLAGS=CAPTIVE /NOLOCAL /NODIALUP -
/LGICMD=SYS$SYSROOT:[USER_AUDIT]CHECKAUDIT
```

Because the captive login command procedure assures that the login is coming from the USER\_AUDIT account via a SET HOST command, the session is audited.

Note that you may also wish to disable batch and network access. For this procedure to work correctly, you must have enabled DECnet proxies with NCP (see Section 7.6).

The CHECKAUDIT.COM procedure verifies that the user is logging into the USER\_AUDIT account.

```
$ ! CHECKAUDIT.COM - ensure that the account is being logged into with
$ ! the session audit account.
$ !
$ IF F$MODE () .NES. "INTERACTIVE" THEN EXIT
$ !
$ ! Verify that the connection originated from the local node and
$ ! from the USER_AUDIT account.
$ !
$ IF F$LOGICAL ("SYS$NODE") .EQS. F$LOGICAL ("SYS$REM_NODE") -
  .AND. F$LOGICAL ("SYS$REM_ID") .EQS. "USER_AUDIT" -
  THEN GOTO OK
$ WRITE SYS$OUTPUT "You may only log into this account with ", -
  "the USER_AUDIT account."
$ LOGOUT
$ !
$ ! When the login has been verified, enable control-Y to
$ ! release the account, invoke the user's LOGIN.COM, and turn
$ ! control over to the user.
$ !
$OK:
$ SET CONTROL_Y
$ IF F$SEARCH ("LOGIN.COM") .EQS. "" THEN EXIT
$ @LOGIN
```

### 5.8.7

#### Other Audit Data

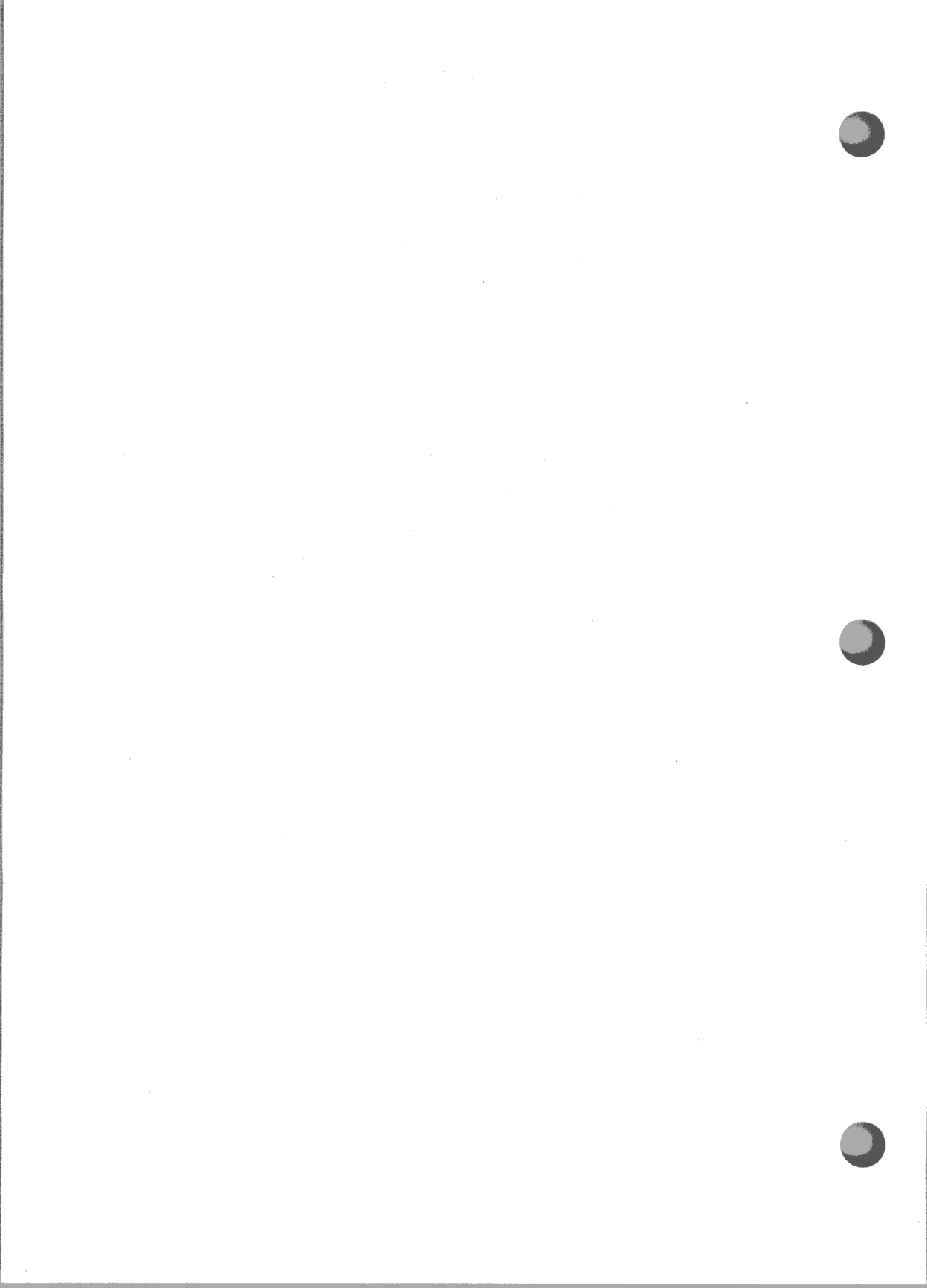
In addition to security alarms, VAX/VMS provides additional data that may be useful in tracking system activity. The system accounting log contains records of all system job terminations. These include all interactive, batch, and network jobs, as well as print jobs and other process terminations. Optionally, activations of all or selected images may also be recorded in the accounting log. Further information on the use of the accounting log is available in the *Guide to VAX/VMS System Management and Daily Operations* and in the *VAX/VMS Accounting Utility Reference Manual*.

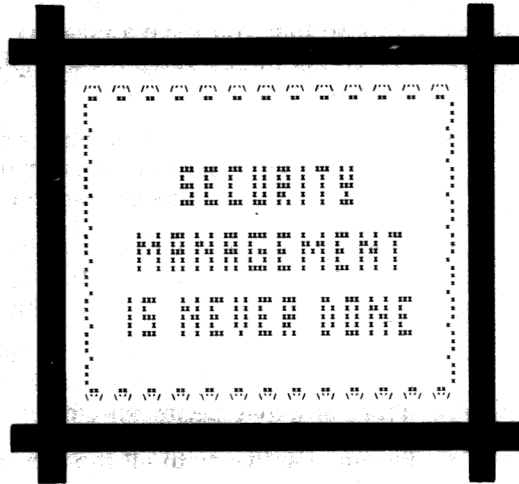
Most network operations, such as mail delivery and access to files from remote network nodes, initiate a network server job for which a log file is created. This log file is normally named NETSERVER.LOG and is located in the default directory of the account under which the job ran. You may be able to use the contents of NETSERVER.LOG when tracking down events that were initiated over the network.

---

### 5.9 Ongoing Tasks

The security manager's work is never done. Even if you have analyzed and implemented the suggestions in this chapter (as well as any in the remaining chapters that apply to your needs) and things appear to be running smoothly, you must plan to review your operation on a regular basis. Timing is exceptionally important when security is at risk. You want to avoid situations where you are so ill-prepared to take action that your poor response time increases the magnitude of the problem. This section describes some activities for your consideration.





ZK-2068-84

## 5.9.1 General Surveillance

Security managers must be ever watchful for indications of trouble. You may find the following checklist directs you to the most productive ways of watching the system and users. Chapter 6 shows you how to apply your observations from these efforts to best interpret signs of attacks on your system or early advance warnings.

- Use the MONITOR IO report to develop an understanding of the normal amounts of I/O on your system at various times. Watch for unseemly changes.
- Keep informed of the images installed on your system. Use the VAX/VMS Install Utility (INSTALL) to look for unexpected additions.
- Use the AUTHORIZE command SHOW on a regular basis watching for usernames that you did not authorize.
- Use the AUTHORIZE command SHOW/PROXY regularly so that you quickly recognize all users that you have authorized. Watch for unexpected additions. Remove any remote users who no longer require access. Institute regular communications on this with your counterparts at the remote nodes.

- Apply the VAX/VMS Accounting Utility on a regular basis so that you have a measure of normal amounts of processing time. Watch for unexplained changes. (The next chapter offers some hints.)
- Regularly check the accounting report produced by the ACCOUNTING for known usernames, unknown usernames, and appropriate hours of system use.
- Develop sufficient understanding of your system's workload that you notice normal (as well as abnormal) processing activity that occurs at unusual hours.
- Observe device allocations with the DCL command SHOW DEVICE on a routine basis so that you will immediately notice any that are unexpected.
- Become familiar with all the recurring types of batch jobs that run on the batch queues and what times they are most likely to run.
- Monitor the protection and ownership of critical files with the DIRECTORY/SECURITY command. Watch for unexplained changes in each.
- Maintain familiarity with the rights database. Keep current listings so that you can recognize identifiers that have been added, or new holders of the current identifiers.
- Remove identifiers that are not in use. Keep the Rights Database current.
- Regularly review the templates that you may be using to set up UAF records. Make any changes necessary.
- Implement security alarms from time-to-time to try to catch browsers.
- Try to break into your user's accounts with some of the obvious password choices.
- Whenever you leave new users responsible for changing their initial passwords, check back at the end of the day to see if you can login with the password you originally assigned. Where necessary, follow up with the user to find out why the change did not occur as requested.

- Try searching unprotected user files for passwords embedded in network access control strings. The password will precede the 3-character terminator ("::). You might also search for the noun "password" and see if any passwords are revealed nearby.
- Check that your users are logging out properly—make physical checks at the end of normal business hours.
- Check that your users have appropriate default protections in place.
- Keep informed about your inventory of magnetic tapes, disks, and program listings. Routinely check that inventory for possible indications that physical security has degraded.
- Keep your office and all important listings locked up.
- Watch for signs of general discontent or instability at your company and tighten security as appropriate. Typical signs are high personnel turnover rates, low morale, economic difficulties, threatened layoffs, increases in attitude problems, plans for new management or takeovers, and so forth.





# 6

## When Your System is Under Attack

---

Recognizing that your system is being attacked is as vital as knowing the techniques to apply to discourage or prevent certain types of attack. From the previous chapters you know that the primary forms of attack are the following:

- Hunting for access lines
- Hunting for passwords
- Attempting a break-in
- Changing or creating UAF records
- Granting/stealing extra privileges
- Introducing software of the Trojan horse type that disguises its true purpose
- Introducing worms in command procedures and programs with the hope they will succeed in propagating themselves into some privileged accounts
- Scavenging disks
- Using a node as a gateway to other nodes

Since it is not always practical or desirable to impose every known security measure to counteract these potential problems, it is actually reasonable policy on some systems to monitor closely for problems, waiting to see indications of trouble before you impose certain restrictions. You are counting on your ability to detect and respond to problems quickly enough that no serious damage occurs. This strategy runs the risk of "closing the barn door after the horse runs away". Thus, it is only suited to systems in the low to low-medium security requirements range.

However, even on systems where security requirements are high and there has been a thorough application of the VAX/VMS security measures, it is still possible to have problems. The goal of this chapter is to help security managers recognize their systems are under attack regardless of the

security measures in effect, and to counsel them on appropriate actions. The discussion of attacks over the network, however, is a very specific type of problem, and that discussion is deferred to Chapter 7.

Security problems will be examined from the point of view of their severity. They will run the gamut from nuisance to potential catastrophe. In the section that discusses actions, the options are discussed from the simplest to the most difficult, which generally corresponds to actions for the least to most severe types of problem.

---

### 6.1 Indications of Trouble

When your system is vulnerable and possibly even under attack, your first indications may come from any of these sources:

- User indications
- Personal observations
- Ongoing auditing applications that you have invoked

The following sections outline the various kinds of indicators of trouble that you may receive.

---

#### 6.1.1 User Indications

Users can provide very helpful indications of problems, as well as many false alarms, too. Your users may contact you and report anomalies such as:

- Files are missing
- Unexplained forms of last login messages appear, such as successful logins that they did not perform or login failures that do not match their experiences
- A sudden inability to log in suggests that their password might have been changed since their last successful login, or some other form of tampering has occurred
- Breakin evasion appears to be in effect and is thwarting their login attempts, yet they experienced no previous login failures

## When Your System is Under Attack

- Reports from the SHOW USERS command indicate that they are logged in on another terminal when they know they are not
- A disconnected job message appears during a login for a process they never initiated
- Software exists in their directories that they did not write
- Unexplained changes have been found in the protection or ownership of their files
- Listings are discovered that were generated under their usernames, yet they never personally requested the listings
- Sudden reduction in the availability of resources such as dialup lines is experienced

Every one of these indications could be signs of serious trouble, if they are valid observations. You should follow up promptly whenever any one of these items is reported to you. You must confirm or deny that the condition exists. If you find the complaint is valid, you must seek a cause and a remedy.

If the report turns out to be unfounded, the user is probably confused, and you or others on the staff may not have done an adequate training job. Remember that poorly informed users can accidentally inflict much damage to systems. Take this opportunity to retrain the user or to assist the user in finding another appropriate source of help.

Whatever action you take, avoid insulting or offending the user; you depend on the user's willingness to report such problems to you in the future. This is important for two reasons. First, all the symptoms mentioned above are very serious indicators of trouble that might be difficult for you to detect without your user's help. Secondly, any user who is having difficulty using the system properly and interpreting this kind of information bears a special kind of watching. You need to know the skill levels of your users so that you can match their access rights and privileges accordingly.

### 6.1.2 Personal Observations

Security managers should be ever alert for a number of conditions that are the precursors of serious problems. Section 5.11.1 suggested some practices that would help you detect problems. The list below illustrates the sort of anomalies you might uncover while performing those recommended tasks. Among the kinds of advance warnings you might obtain (other than from some of the formally designed tests that Section 6.2 presents) are:

- A user appears on the SHOW USERS report that you know could not be logged in at the moment.
- You observe an unexplained change in the system load.
- You discover some media or program listings are missing, or obtain other indications that physical security has degraded.
- Your locked file cabinet has been tampered with and the list of authorized users has disappeared.
- You find strange software in the system executable image library [SYSEXE].
- You observe unfamiliar images running when you examine the SHOW SYSTEM report.
- You observe usernames that you did not authorize, and then when you examine the listing that AUTHORIZE produces with the SHOW command you find that somehow the users have been authorized.
- You uncover PROXY users that you never authorized.
- The accounting report reveals unusual amounts of processing time expended recently, suggesting outside access.
- You observe unexplained batch jobs on the batch queues.
- You observe device allocations with the SHOW DEVICE command that are unexpected.
- High personnel turnover rates at the company, low morale, economic difficulties, or attitude problems seem rampant.
- You observe normal processing activity but at unusual hours.

## When Your System is Under Attack

- The UIC-based protection or ACLs change on critical files. Identifiers are added, or holders of identifiers are added to the rights database.

All these conditions warrant further investigation. Some indicate that you already have a problem, while others may have innocuous explanations, or may indicate that serious problems could arise unless you take preventive action.

---

## 6.2 Routine System Surveillance

VAX/VMS provides a number of mechanisms that allow systematic surveillance of the activity in your system. Proper use of such mechanisms should help alert you to problems, and allow you to intervene in a timely way. This section describes the most important ones.

---

### 6.2.1 Accounting Log

Accounting logs that you can generate with the VAX/VMS Accounting Utility can provide a number of helpful indications of problems, possibly before the problems get out of control. You should check your logs for the following items:

- Unfamiliar usernames
- Unfamiliar patterns of use, such as activity that is unusual for a particular time of day or day of week
- Use of an abnormal quantity of resources
- Unfamiliar sources of login, such as network nodes or terminals

By remaining vigilant in these areas, you improve the likelihood of early detection of threats.

---

### 6.2.2 Security Auditing

With the DCL command SET AUDIT you can enable a variety of alarms. The following security alarm features are presented in rough order of decreasing priority and increasing cost. Section 4.9.2 presents an example of how to establish a security alarm.

- 1 Enable security auditing for LOGFAIL and BREAKIN. This is the best way to detect probing by outsiders (and insiders looking for accounts). Sites with any level of security awareness should enable these.
- 2 Enable security auditing for LOGIN. Auditing successful logins (possibly only from the more suspicious sources, like REMOTE and DIALUP, but perhaps from all) provides the best way to track which accounts are being used. This claim derives from the fact that an audit record is written before users logging in on a privileged account can disguise their identity.
- 3 Enable the FILE=FAILURE type of security audit. This technique audits all file protection violations and is an excellent method of catching probers.
- 4 Apply ACL-based file access auditing to detect WRITE access to critical system files. The most important files to audit, because of their security implications if tampered with, are shown in Table 6-1.

The list in Table 6-1 is not all-inclusive, but serves to illustrate the more critical system files that you could choose to audit. Attacks on these files constitute direct attacks on the security mechanisms of VAX/VMS. A would-be intruder could also plant a Trojan horse in any other command file or executable program used by system managers or operators. You may want to audit only successful access to detect penetrations, or you may want to audit access failures as well, to detect probing.

Note that some of these files are written during normal system operation. For example, SYSUAF.DAT is written during each login, and SYSMGR.DIR is written when the system boots.

## When Your System is Under Attack

- 5 Consider auditing accesses to files that contain the most critical data belonging to your installation.
- 6 Audit use of privilege to access files (perhaps only writing, or perhaps all forms of access). Implement the security audit with FILE=(SYSPRV,BYPASS,READALL,GRPPRV). Note that this class of auditing can produce a large volume of output, because privileges are often used in normal system operation for such tasks as mail delivery, operator backups, and so forth.

**Table 6-1 System File Candidates for ACL-based File Access Auditing**

Device and Directory	File Name
SYS\$SYSTEM	SYS.EXE
	F11BXQP.EXE
	LOGINOUT.EXE
	DCL.EXE
	JOBCTL.EXE
	JBCSYSQUE.DAT
	SYSUAF.DAT
	NETUAF.DAT
	RIGHTSLIST.DAT
	STARTUP.COM
	SECURESHR.EXE
SYS\$LIBRARY	SYSTARTUP.COM
SYS\$MANAGER	VMSIMAGES.DAT
SYS\$SYSROOT	[000000]SYSEXE.DIR
	[000000]SYSLIB.DIR
	[000000]SYSMGR.DIR

### 6.3 Handling a Security Breach

Typically, there are four phases that security managers experience while handling a security breach, whether the breach actually occurred or merely was attempted:

- 1** Detection—discovering that there is a problem
- 2** Identification—determining who the perpetrator is
- 3** Prevention—preventing further violations of security
- 4** Repair—repairing whatever damage was done

Your experience during these phases differs for attempted and successful breaches, as revealed in the rest of this section.

---

### **6.3.1 Unsuccessful Breakin Attempts**

Unsuccessful breakin attempts refer to situations where you discover that someone has been attempting to guess passwords, or browsing through the file structure.

---

#### **6.3.1.1 Detection of the Unsuccessful Breakin Attempt**

You generally accomplish this detection through one or more of the following occurrences:

- Reports from your users about unexplained login failures
- Unusual system activity or unavailability of dialup lines
- Security alarms for login failures, breakin detection, and file protection violations

---

#### **6.3.1.2 Identifying the Perpetrator**

If you have file auditing enabled, identifying the browser is fairly straightforward. If, however, you discover that rummaging through files is initiated from another node in the network, you must inspect the FAL logs that correspond to the times of the protection violations. Eventually, you will likely have to coordinate with the security manager on the remote node.

Identifying the perpetrator who is guessing passwords is considerably more difficult, especially when the source is anonymous, as it is from a dialup line. Usually, you must trade identification against prevention. Often the only way to positively identify an outsider who is attempting to enter the system will require that you sacrifice prevention and permit further attempts, while you hope to establish the perpetrator's identity. This tradeoff of identification against prevention is further discussed in the Section 6.3.2.2.



### 6.3.1.3 Prevention of Breakin Attempts

The prevention phase for this kind of attack involves (1) preventing the would-be intruder from actually gaining access to the system, and (2) making future attempts more difficult. You should focus your prevention efforts towards eliminating password guessing and file browsing.

#### Password Guessing

To reduce the opportunities for successful password guessing, take the following steps:

- Make certain your users employ adequate passwords. Warn your users that someone is attempting entry; this often provides a lot of motivation. Consider across-the-board enforcement of the use of the password generator. See Section 5.2.7.6.
- Consider enabling system passwords on the points of entry. At a minor inconvenience to your users, you have about the best protection against further probing available. (If you already had a system password enabled, change it!) See Section 5.2.7.3.
- Enable auditing of successful logins, to catch the event if the intruder succeeds in getting in. See Section 6.2.2.

#### File Browsing

To reduce the opportunities for successful file browsing, take the following steps:

- If you can identify the culprit, discuss the event with that user. The realization that they have been caught is often effective in deterring browsers from future activity.
- Warn your users about the importance of adequate protection of their files, and consider inspecting the protection of your users' files.
- If file browsing from other nodes in the network becomes a persistent problem, consider eliminating the default FAL account and authorizing individual users through proxy login accounts. See Section 7.6.

---

**6.3.1.4 Repair After an Unsuccessful Breakin**

Repair of files or data structures is not necessary in this class of breakin, since no damage has been inflicted on them. The losses that you have suffered cannot be repaired. Typically, you have lost information to the browser. The value of the information determines the extent of the loss.

---

**6.3.2 Successful Breakin Attempts**

A real security breach can be a traumatic event, akin to a burglary of your home or office, and will require considerable time and effort on your part, and on the part of your users, to overcome.

---

**6.3.2.1 Detection of Successful Breakin Attempts**

You accomplish detection by any of the measures described thus far in this chapter. The event can be very obvious, as when you discover that critical files have been destroyed. However, it can also be very subtle, as when you begin to realize that an unknown user has been using your system for the past week.

---

**6.3.2.2 Identification of Breakin Perpetrator**

Identification is often the most difficult part of handling a breakin. First, you must establish whether the perpetrator is an insider or an outsider (that is, one of your authorized users or someone else). Making this level of identification is fairly important, since it determines the nature of the preventative measures that you will take, which are described below. However, even this basic distinction between insiders and outsiders may be difficult to achieve. First, gather and analyze all the information you have available from your audit and accounting logs, from the nature of the penetration, from the perpetrator's apparent knowledge of your installation, and so forth.

### **Tradeoff Between Identification and Prevention**

You will likely have to make a tradeoff between a more positive identification of the intruder and preventing future attacks. Often, the data available initially does not allow complete identification. If it is important to identify the perpetrator, you will often find it necessary to permit continued breakins while you analyze the breakin activity. Stepped up auditing is your most important tool under these circumstances. Consider planting traps in system procedures that are under your control (such as SYLOGIN.COM) to obtain additional useful information. Also redouble your system backup efforts to permit easier recovery if files become damaged.

### **Identification of Outsiders**

Identifying outsiders is particularly difficult, especially if they use any switched forms of communication (such as dial-up lines, public data networks, and so forth). DECnet-VAX provides sufficient features to allow you to trace the activity through the network back to the source node. If a local terminal is involved, physical surveillance may provide the answer.

However, when a switched connection is involved, you must realize that one of the major computer security problems you face is the telephone system itself. Tracing a telephone or public data network connection is a very time consuming and frustrating experience. Calling up the telephone company and having a call traced 10 minutes later is an event that occurs only in grade B movies and television detective shows. Chasing an intruder through the telephone system is likely to take months, and it will definitely require the assistance of law enforcement authorities. Considerations for individual privacy dictate that you will make little progress with telephone companies on your own. Finally, the advent of independent long-distance telephone services compounds the problem by increasing the number of non-cooperating organizations you could encounter.

As a result, identifying an outside intruder is usually worthwhile only when you have sustained substantial damage, and you can measure that damage in terms of money—generally a lot of money at that. In many cases, you profit more if you simply concentrate your energy on preventing future recurrences of the problem.

### 6.3.2.3

#### Prevention of Breakin Attempts

The steps you must take to secure your system after a breakin depend in large part on the nature and source of that breakin. This section describes them in general order of priority.

- Secure your authorization files. The best practice is to restore SYSUAF.DAT, NETUAF.DAT, and RIGHTSLIST.DAT from backups, to undo whatever modifications the perpetrator might have made. Alternatively, generate listings of the files and inspect them closely, looking for such things as improper entries, additional privileges, and changed UICs. If you are unsure of when the UAF might have first been modified, inspect it carefully, regardless of whether you are using a backup copy or proceeding with the existing one.
- Change passwords. DIGITAL believes that the password encryption algorithm used in VAX/VMS is uninvertible. However, the penetrator may have discovered passwords through other means (by browsing files, or from other individuals, or from other nodes in the network). Also, the penetrator may have changed the passwords on some existing, but little-used accounts, to convert them for personal use. At a minimum, change passwords on all privileged accounts. Ideally, change the passwords on all accounts and have your users appear in person to learn their new passwords. **Do not** take the shortcut of using the same new password for all accounts.
- Clean up your system software. A sophisticated penetrator may have planted "trap doors" in the system to provide future access even though you have taken the obvious steps of securing your system. Therefore, you may have to restore selected components of the VAX/VMS software from backups or from your VAX/VMS distribution kit. If the intruder was an outsider, the only critical component is LOGINOUT.EXE, since it validates all entries to the system.
- However, insiders (that is, your authorized users) can make use of a wide variety of trap doors in the executive (SYS.EXE), the file system (F11BXQP.EXE), DCL, and so forth. Furthermore, the penetrator may have planted a Trojan horse in any piece of software or command procedure likely to be used by a privileged user. Thus, complete assurance of a "clean" system requires a rather wholesale restoration of files from backups. An alternate strategy is to restore trustworthy copies of the obvious

targets of attack, and rely on increased auditing for a period of time to catch suspicious events.

- Tighten security. Finally, consider implementing additional security features, such as system passwords, password generation, increased auditing, more stringent file protection, and so forth, to prevent a recurrence.

---

### 6.3.2.4 Repair After a Breakin

Repair consists in large part of a continuation of the activities already described. You must restore corrupted files and so forth. You must decide whether it is appropriate to simply do a wholesale restore of your system's data, or repair problems as they are discovered by you and your users. Remember that you must look not only for file corruption, but also for modifications to file protection that would have created worm holes, and for Trojan horses that were unknowingly introduced into the system and still reside there.

---

## 6.4 Summary

This chapter presents techniques for detecting and recovering from security problems. Besides acquiring an understanding of how to handle such problems, you likely developed an appreciation of the consequences of a breakin. Such appreciation is usually a strong motivator to prevent the possibility of the occurrence in the first place.



# 7

---

## Security for a DECnet Node

Security in a networking environment is even more sensitive than security in a single system environment. It is also harder to achieve because of the dual problems posed by operational complexities and the decentralization of control that commonly exists in networks. The larger the network, the more difficult the problem of establishing control and communication between the security managers of the numerous nodes becomes. When one manager has total control and decision-making authority, the consistency and integrity tend to be higher. All this is not to say that consistency and operational integrity cannot be achieved, but the dedication required of all participants should not be underestimated.

This chapter provides direction on how security managers can improve the security in their networks. There are some critical features that can be implemented. However, this chapter cannot dictate operational policy between various nodes to maintain proper security. This task is even more important to overall security than individual node operations; it must be addressed and refined regularly. Security managers should plan from the outset to dedicate significant time to this part of their responsibilities.

There are limitations in the degree of security any networking site can expect to achieve. Such limitations are caused by limitations currently present in the networking technology. No doubt, future innovations will attempt to overcome these limitations, but in the meantime, it is important to be realistic about just how secure you can hope to make your networking system. Knowing where problems could arise can help you to avoid operations that could increase the security exposure in your network. This chapter should help you to recognize these problem areas and to adjust your operations accordingly.

This chapter assumes the reader has read and mastered the information in the *Guide to Networking on VAX/VMS*.

---

## 7.1 The Reference Monitor in a Network

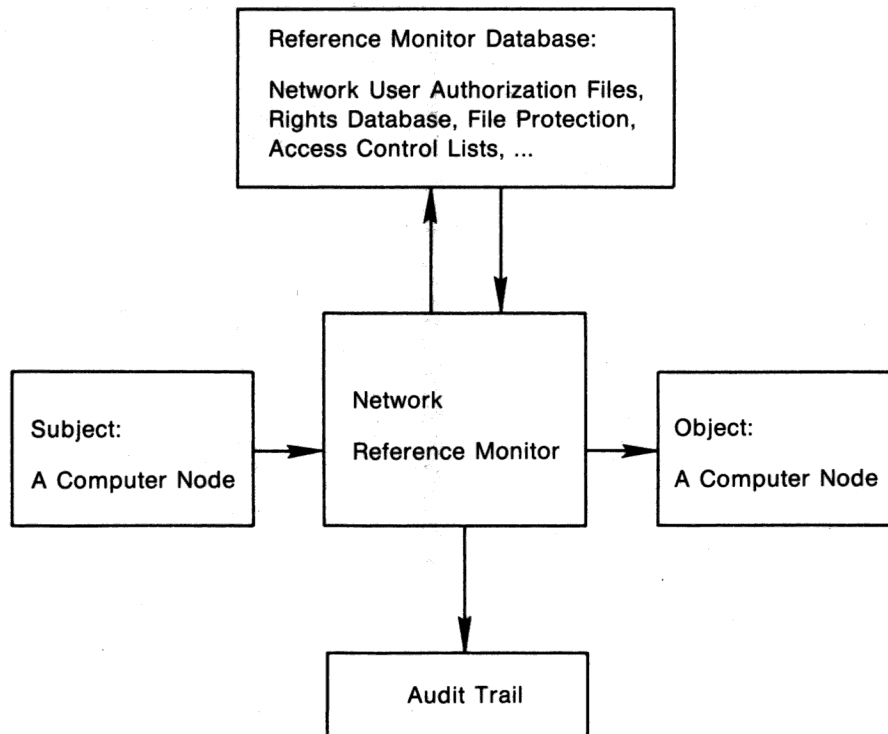
Chapter 2 introduces the concept of the reference monitor. This concept also applies to security in a network of interconnected computer systems. This section first extends the reference monitor concept to the network environment, then summarizes the special considerations that apply in a network, and finally makes the connection between the abstract components of the reference monitor concept and the real elements of a DECnet-VAX network.

When you think about gaining access to information in a network, you might begin by extending the reference monitor concept while temporarily disregarding the presence of the network. There is a subject on one computer in the network, an object on another, and you can imagine a network reference monitor that grants the subject access to the object, refers to an authorization database, and develops the required audit trail. Figure 7-1 shows this simplified view of secure access in a network environment.

To a certain extent, the network security mechanisms that DIGITAL employs conform to the abstract model depicted in Figure 7-1. However, the reality is somewhat more complex. Therefore, it is worth examining a slightly more elaborate model. In reality you must consider two computer systems; one that holds the subject that needs to gain access and a second one that holds the object that is the target of the access attempt. Note that each computer must have its own implementation of the reference monitor abstraction if there is to be security in the resulting network. Since the individual reference monitors each deal with subjects and objects that are local to them, there will be a phantom object on the system with the real subject (the source machine) and a corresponding phantom subject on the system with the real object (the target machine). The resulting configuration resembles Figure 7-2.

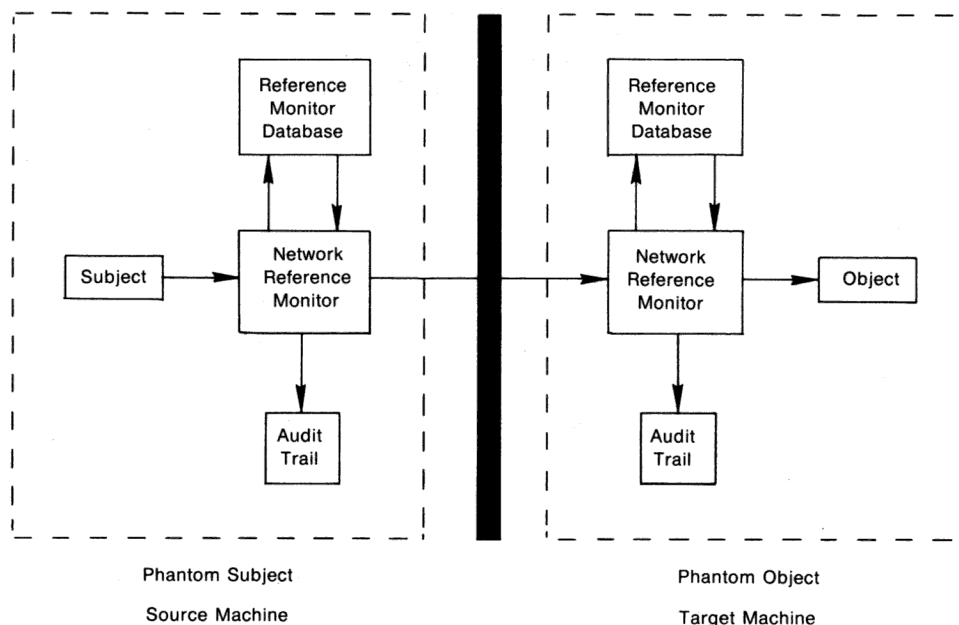


**Figure 7-1 Simple Diagram of Reference Monitor in a Network**



ZK-2018-84

**Figure 7-2 Advanced Diagram of the Reference Monitor in a Network**



ZK-2038-84

Given this view of secure access to information in a network, there are three critical requirements for achieving security in a network environment:

- 1 There must be a correspondence between the real subject on the source machine and the phantom subject on the target machine. This correspondence must be managed by the two reference monitors and must be consistent with the security policy intended on the target machine (which, is ultimately responsible for protecting the object).
- 2 The authorization database on the target machine must express an access authorization for a phantom subject that corresponds to the correct real subject on the correct source machine.

- 3 There must be a protected means of communication between the two reference monitors (source and target) so that the correspondence between real and phantom subjects can be reliably established and authenticated.

VAX/VMS provides mechanisms to help meet each of the first two requirements. Mechanisms for meeting the third requirement are discussed in Section 7.1.3.

---

### 7.1.1 Establishing Subject Correspondence

VAX/VMS and DECnet-VAX provide several mechanisms for establishing a correspondence between a subject or process on a source node and another on a target node. Essentially, the default account mechanisms allow any subject on any node to be placed in correspondence with a default subject on a target node. This subject can in turn gain access to objects on behalf of a requesting subject and return the required information. Note that the fact that any subject can be placed in correspondence with a default subject on a target node implies little selectivity or control in the establishment of the correspondence.

At the other extreme is the use of explicit or username /password access control on the establishment of a subject at the target node. This mechanism restricts access to those objects accessible to the named user, but has the side effect of causing users' passwords to flow about the network—often without effective protection.

VAX/VMS offers a final option for establishing the correspondence between subjects—proxy accounts. Section 3.2.2 describes proxy accounts. The proxy option requires the target reference monitor to maintain a table of source subjects (by user name and node name) and the corresponding local (target) user names. Then each request from a subject on a source node will be mapped into the creation of a subject representing the corresponding target user. This mechanism offers the explicit control associated with username/password control but more adequately protects the passwords.

---

### 7.1.2 Specifying Authorizations

The approach used to specify authorization for access to objects depends somewhat on the mechanism for establishing correspondence between subjects. The various default account mechanisms essentially create anonymous subjects on the target node. As a result, objects that are to be made accessible to a default account must permit the WORLD user category full access, which leaves the object unprotected.

If either explicit access control or proxy access is used to establish correspondence between subjects, the authorization can be granted to the target subject selected by the username or proxy. In this case, the full range of VAX/VMS authorization mechanisms can be used.

---

### 7.1.3 Protecting Communications

It should be clear that the security of network operations depends in large measure on the ability of source and target reference monitor mechanisms to communicate in a private and authentic manner. An intruder must not be allowed to observe passwords or to pretend to be a source node that has been granted proxy access.

Unfortunately, the realm of protected communications is at present outside the domain of VAX/VMS security. Users can achieve such protection either through physical protection of the communication lines or by protecting these lines by using encryption. Mechanisms for physical protection (conduit or perhaps fiber optics) and encryption are available from a variety of third party vendors.

Readers should not assume that network security measures such as conduits and/or encryption are needed only in high-security environments. Rather, many communication media, including most local area networks like Ethernet, are so unprotected that an intruder or authorized user with a personal computer can easily read or forge any information that flows over the network.

---

#### 7.1.4 Summary of VAX/VMS Network Security and the Reference Monitor

To summarize, VAX/VMS provides, especially through the proxy mechanism, a vehicle for extending user authentication and authorization over a network in a secure, natural and consistent manner. However, from a system point of view, the network security mechanisms, are no better than the protection of the underlying communications. This latter point can be critical in relatively open networks that process sensitive information.

---

### 7.2 DECnet-VAX Accounts

The DECnet-VAX accounts permit certain types of access to your system from remote nodes without requiring them to specify the account and password information. Instead, this information is specified in the DECnet-VAX executor and object databases. Like all accounts, these are controlled through the system authorization file using techniques similar to those used for captive user accounts.

You should consider the following general guidelines whenever you set up accounts for DECnet-VAX use. Detailed examples are given later.

- DECnet-VAX currently has no requirement for a privileged default account, and you should not provide one. In addition, only create a default account for objects when required, rather than setting one up for the executor. DECnet-VAX requires a privileged account for the local use of the Network Management Listener (NML), but you need not set it up as any form of default. It can be specified in a SET EXECUTOR command when required.
- UICs of the network nonprivileged accounts should be unique for each group and user. Furthermore, the group code must exceed the system UIC group number to avoid granting the user the SYSTEM user category for file access. (You can ensure this by using group codes that are greater than the SYSGEN parameter MAXSYSGROUP).
- Keep the privileges for DECnet-VAX accounts to a minimum. Typically this means you would only give TMPMBX and NETMBX to nonprivileged accounts.

- Maintain the secrecy of passwords for the DECnet-VAX accounts; they need not be known to users of your node or other nodes. Once the password is defined in the authorization file and the DECnet-VAX databases, there will be no need to specify the password.
- Set up the accounts with the following AUTHORIZE qualifiers: /FLAGS=(DISCTLY,CAPTIVE,LOCKPWD), /NOINTERACTIVE, and /NOBATCH.
- The account for the FAL object should have a group code in its UIC that differs from every other account in the system, including accounts for other DECnet objects. Note that if you have a task object defined, you can prevent outsiders from running arbitrary command procedures on your system if you make the UIC group of the task object different from the UIC group of the default FAL account.
- The member number of the owner UIC of the default directory for the FAL account should be different from the member number of the owner UIC of the FAL account. This ensures that READ and/or WRITE access is permitted but CONTROL access is not. Without CONTROL access, a FAL account user cannot change the protection of the directory.
- For any account that does need not to support remote file access, you might find it useful to place the following command in the account's login file:

```
$ FAL$COMMAND == LOGOUT
```

However, this technique only works when the FAL object is defined to execute the command procedure SYS\$SYSTEM:FAL.COM. Since the DECnet-VAX default defines the FAL object to call FAL.EXE directly instead, you will probably need to execute the following DECnet-VAX command:

```
NCP> DEFINE OBJECT FAL FILE FAL.COM
```

In fact, this technique can be used with user accounts as well as with the DECnet-VAX accounts, to prohibit remote file access by logging out any user who attempts it. Note that this technique shuts off only the remote file access, but allows access to other objects. This is preferable to specifying /NONETWORK in the user's UAF, since that would deny all DECnet use.

---

### 7.3 The DECnet-VAX Database

The DECnet-VAX node and circuit databases control how other computers are allowed to connect to your computer. Since a computer connection permits automated assaults on both your own security and that of any other computer in the network, it requires as strict control as a user account.

To promote the security of the databases, you should observe the following guidelines:

- Define receive and transmit passwords for all nodes in the database. The receive password defined for a node need be known by the manager of that node only if that node can be adjacent. Wherever possible, the transmit and receive passwords should be different, and not obvious. (However, some operating systems do not permit this.)
- Verification must always be enabled on any circuit that goes outside a locked computer room, or goes to a machine with a different security environment. This is necessary to prevent the node adjacent to you from pretending to be another computer, and thus, from intercepting mail or circumventing a connection check for the originating node. This is particularly important whenever proxy logins are permitted.
- Do not define default access rights in the database for external nodes. (A possible exception to this would be for a dedicated front end or server).
- In general, backup synchronous dialup should not be enabled for autoanswer. Systems that have incoming dialup for production purposes should control which nodes can connect.

---

## 7.4 Network Usage

The type of use that may be made of the network is restricted in many foreign countries, either by law or by the contract with the major communications division of the government known as the PTT. While this is not strictly a security issue, anyone responsible for security should be aware and take steps to conform to these regulations. For example, several countries have laws to protect personal data from misuse, and these normally include restrictions on moving such personal data across country boundaries. This would obviously include moving or remotely accessing personnel databases, but it might be interpreted to include forwarding of job applications, and so forth. Furthermore, Germany has a law intended to promote self-sufficiency in computer resources that forbids transmitting data for processing outside the country.

Similarly, in some European countries either law or the PTT contract forbids anyone but the PTT from providing data transmission as a service to customers. And, in Germany it is not legal to route data between the X.25 network and the leased line network.

Security managers should attempt to understand all such regulations and to ensure that their sites are in compliance.

---

## 7.5 Specifying DECnet Object Accounts

The following section describes parts of a command file for NCP that sets up some of the usual objects with explicit accounts, the authorization file entries for the accounts, and the associated login command files. Note that the account used for the FAL object is different from the others. With explicit accounts for all required objects, there is no need for executor default accounts.

Figure 7-3 illustrates how the definition for the DECNET and FAL account might appear in the object database.



---

**Figure 7-3 Definitions in the Network Object Database**

---

```
!
! For object FAL account is special
DEFINE OBJECT FAL -
  NUMBER 17-
  FILE SYS$SYSTEM:FAL-
  USER FAL PASSWORD ABCXYZ
!
! Allow network information
DEFINE OBJECT NML -
  NUMBER 19 -
  FILE SYS$SYSTEM:NML -
  USER DECNET PASSWORD XYZABC
!
! Allow MAIL
DEFINE OBJECT MAIL -
  NUMBER 27 -
  FILE SYS$SYSTEM:MAIL -
  USER DECNET PASSWORD XYZABC
!
! Allow PHONE
DEFINE OBJECT PHONE -
  NUMBER 29 -
  FILE SYS$SYSTEM:PHONE -
  USER DECNET PASSWORD XYZABC
!
```

---

Figures 7-4 and 7-5 present typical entries in SYSUAF.DAT for these two accounts.

**Figure 7-4 UAF Record for FAL Account**

```

Username: FAL                               Owner: OFFICE
Account: NETNODE                             UIC: [301,303] ([DECNET3,FAL])
CLI: DCL                                     Tables:
Default: DOCD$:[FAL]
LGICMD: FALLOG.COM
Login Flags: Disctly Defcli Lockpwd Captive
Primary days: Mon Tue Wed Thu Fri Sat Sun
Secondary days:
Primary 000000000001111111112222 Secondary 000000000011111111112222
Day Hours 012345678901234567890123 Day Hours 012345678901234567890123
Network: ##### Full access #####          ##### Full access #####
Batch: ----- No access -----          ----- No access -----
Local: ----- No access -----          ----- No access -----
Dialup: ----- No access -----          ----- No access -----
Remote: ----- No access -----          ----- No access -----
Expiration: (none) Pwdminimum: 6 Login Fails: 0
Pwdlifetime: (none) Pwdchange: 24-JAN-1985 17:19
Last Login: (none) (interactive), 12-MAR-1985 16:49 (non-interactive)
Maxjobs: 0 Fillm: 16 Bytlm: 12480
Maxacctjobs: 0 Shrfillm: 0 Pbytlm: 0
Maxdetach: 0 BIOLm: 12 JTquota: 1024
Prclm: 0 DIOLm: 6 WSdef: 180
Prio: 4 ASTlm: 16 WSquo: 200
Queprio: 0 TQEIm: 10 WSextent: 0
CPU: (none) Enqlm: 20 Pgflquo: 25600
Authorized Privileges:
TMPMBX NETMBX
Default Privileges:
TMPMBX NETMBX

```

**Figure 7-5 UAF Record for DECNET Account**

```

Username: DECNET                      Owner: DECNET
Account:  NETNODE                     UIC:   [300,300] ([DECNET2,DECNET])
CLI:      DCL                         Tables:
Default:  DOCD$:[DECNET]
LGICMD:   LOGIN.COM
Login Flags:  Disctly Defcli Lockpwd Captive
Primary days: Mon Tue Wed Thu Fri Sat Sun
Secondary days:
Primary 00000000001111111112222 Secondary 00000000001111111112222
Day Hours 012345678901234567890123 Day Hours 012345678901234567890123
Network: ##### Full access #####      ##### Full access #####
Batch:  ---- No access ----            ---- No access ----
Local:  ---- No access ----            ---- No access ----
Dialup: ---- No access ----            ---- No access ----
Remote: ---- No access ----            ---- No access ----
Expiration: (none) Pwdminimum: 6 Login Fails: 0
Pwdlifetime: (none) Pwdchange: 24-JAN-1985 17:21
Last Login: (none) (interactive), 23-MAR-1985 16:49 (non-interactive)
Maxjobs: 0 Fillm: 16 Bytlm: 12480
Maxacctjobs: 0 Shrfillm: 0 Pbytlm: 0
Maxdetach: 0 BIOfm: 12 JTquota: 1024
Prclm: 0 DIOfm: 6 WSdef: 180
Prio: 4 ASTlm: 16 WSquo: 200
Queprio: 0 TQElm: 10 WSextent: 0
CPU: (none) Enqlm: 20 Pgflquo: 25600
Authorized Privileges:
TMPMBX NETMBX
Default Privileges:
TMPMBX NETMBX

```

Creating and maintaining a FAL account is sensible at small sites where operations are generally friendly and security requirements are low to moderate. However, some sites may decide to eliminate the FAL account. Good candidates for FAL account elimination are either sites with high security requirements or sites with such large numbers of users that it becomes increasingly difficult to recognize all the intended users. When there is no FAL account, random, unknown users are unable to gain entry to the system. To specifically control which users gain access, these sites might choose to establish one or more proxy accounts for specific purposes.

---

## 7.6 Proxy Logins

From the description in Section 3.2.2 you know that when you encounter situations where users on different nodes or in different groups want to share files on your system and you are reluctant to give out passwords or to set the directory and file protection to WORLD:RWE, you authorize proxy access as an alternative. With proxy logins, there is no need for passwords to be embedded in commands to copy a file. Also, there is no need for a file's protection code to be set to allow the WORLD category of users READ access to transfer a file. The user simply issues the following form of the DCL command COPY:

```
COPY remotenode::file-spec file-spec
```

---

### 7.6.1 Setting Up Proxy Logins

Two utilities are used to setup proxy logins: AUTHORIZE and NCP. You might want to create a command procedure to assist you in implementing proxy access through a step-by-step approach.

For example, the command procedure could provide the following functions:

- Check if proxy is turned on for your system
- Create a proxy account for sharing files
- Add a user to access a proxy account
- Remove a user from access to a proxy account
- List users authorized to access a proxy account

---

#### 7.6.1.1 Using the VAX/VMS Authorize Utility

To set up proxy logins in lieu of using a command procedure, use AUTHORIZE to create or modify the NETUAF.DAT file containing the remote-node::user to local-user mapping. Briefly, the commands related to establishing the proxy database are as follows:

- CREATE/PROXY
- ADD/PROXY node::remoteuser localuser
- LIST/PROXY
- SHOW/PROXY node::remoteuser
- SHOW/PROXY \*
- REMOVE/PROXY node::remoteuser

---

#### 7.6.1.2 A Proxy Account

When you want to set up a proxy account on your node for use by one or more users at other nodes, you must perform the following steps:

- Decide on the purpose of the account. Decide the name of the local account and which foreign users will be admitted.
- If the local account does not exist, create it with AUTHORIZE; if the account does exist, examine it to ensure it is adequately restricted. Proxy accounts should be restricted so that they prohibit interactive users and batch jobs, which is to say they should only permit network logins.
- Review the privileges on the account. Generally, you should avoid granting privileges to proxy login accounts. This practice provides a kind of shield between systems in a network in the event one node is penetrated. The fact that proxy logins only provide admittance to nonprivileged accounts at other nodes should help to contain the extent of damage if a penetration occurs on one system in the network.
- If the network user authorization file does not already exist, create it with the AUTHORIZE command CREATE/PROXY.

- Add as many network user authorization records as necessary with the AUTHORIZE command ADD/PROXY. (Exercise caution when you agree to authorize users, since you may not personally know all the users. Ideally, you should receive a formal request from the security manager at the remote site.)
- Check the default protection on the directory and customize it as necessary.
- Examine any command procedure used at login time and specified by /LGICMD. Make certain that it follows the recommendations in Section 5.8 for login command procedures in captive accounts. It should reside in a well-protected directory owned by a user other than the owner of the proxy account, and it should prohibit WRITE access for those who use the account.
- Notify the security manager at the remote node what users from that node have been authorized for access to your node.

In Figure 7-6 the security manager at the node WALNUT wants to create a general access account called GENACCESS. At the same time the manager wants to take steps to allow proxy logins by three users from the node BIRCH: KMAHOGANY, PSUMAC, and WPINE as well as two users from the node WILLOW: RDOGWOOD and WCHERRY. Assume no network user authorization file currently exists.

Remember that AUTHORIZE performs certain automatic maintenance functions on the NETUAF.DAT proxy login file. Whenever the username changes through a RENAME or COPY command, the associated change is made in the NETUAF. Similarly, when you remove an account from SYSUAF.DAT, all entries for which there is a matching local username are removed from NETUAF.DAT.

**Figure 7-6 Example of a Proxy Account**

```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF> ADD GENACCESS /PASSWORD=WHYNADGUM/UIC=[236,043] -
_/DEVICE=STAFFDEV/DIRECTORY=[GENACCESS] -
_/OWNER="Security Mgmt"/ACCOUNT=SEC -
_/FLAGS=(DISWELCOME,DISNEWMAIL,GENPWD,DISMAIL) -
_/NOBATCH/NOINTERACTIVE/MAXDETACH=8 -
_/LGICMD=LOGIN/MAXACCTJOBS=10
user record successfully added
identifier for value:[000236,000043] added to RIGHTSLLIST.DAT
UAF> CREATE/PROXY
UAF> ADD/PROXY BIRCH::KMAHOGANY GENACCESS
record successfully added to NETUAF.DAT
UAF> ADD/PROXY BIRCH::PSUMAC GENACCESS)
record successfully added to NETUAF.DAT
UAF> ADD/PROXY BIRCH::WPINE GENACCESS
record successfully added to NETUAF.DAT
UAF> ADD/PROXY WILLOW::RDOGWOOD GENACCESS
record successfully added to NETUAF.DAT
UAF> ADD/PROXY WILLOW::WCHERRY GENACCESS
record successfully added to NETUAF.DAT
UAF> SHOW/PROXY *::*
Node      Remote User    Local User
BIRCH    ::KMAHOGANY    GENACCESS
BIRCH    ::PSUMAC       GENACCESS
BIRCH    ::WPINE        GENACCESS
WILLOW   ::RDOGWOOD     GENACCESS
WILLOW   ::WCHERRY      GENACCESS
UAF> EXIT
{messages}
$ DIRECTORY/SECURITY SYS$STAFF:[000000]GENACCESS.DIR

$ DIRECTORY/SECURITY SYS$STAFF:[GENACCESS]LOGIN.COM
```

### 7.6.1.3

#### Using the VAX/VMS Network Control Program Utility

You use NCP to control the overall use of proxy login with respect to the executor node and network objects. There are three access control parameters, each of which may have any of the values shown in Table 7-1.

**Table 7-1 Access Control Parameter Values**

Parameter	Meaning
NONE	Neither inbound nor outbound proxy access is allowed.
INCOMING	Proxy access is allowed on connect requests originating remotely, that is, inbound connections will use proxy; outbound connections will not.
OUTGOING	Proxy access is requested on locally initiated logical links but is disabled on connections from remote nodes.
BOTH	Both inbound and outbound proxy access are allowed.

Whenever possible, use the value OUTGOING instead of NONE since it will cause the username, and not the PID (process id), to be shown in the NCP display SHOW LINKS. Furthermore, use the value BOTH if you are allowing proxy access to your system.

The control parameters are found in the EXECUTOR and OBJECT databases. They are each part of the CHARACTERISTICS display that you can generate with the following command:

```
$ RUN SYS$SYSTEM:NCP SHOW CHAR OBJ FAL
```

The EXECUTOR database contains the DEFAULT PROXY parameter. This is the parameter used to supply values for the other parameters whenever they are not explicitly set up for a given node or object. The purpose of this parameter is to make it easy to set up the DECnet-VAX configuration database.

Proxy access will not function for nodes that have any privileged or nonprivileged access control specified (parameters NONPRIVILEGED (or PRIVILEGED) USER, PASSWORD, and ACCOUNT). This is because the concept of outbound proxy access conflicts with the concept of default outbound access control strings. This conflict occurs on the destination node. When a connect message containing non-null access control strings is received, the receiving node has no way of knowing whether those strings were specified explicitly by the user or were defaults provided by the source-node operating system;



whenever access control strings are passed in the connect message, they are used and proxy access is inhibited.

The USER, PASSWORD, and ACCOUNT parameters are vestiges of an earlier design and should rarely be used. They are still needed if default access is to be provided to nodes that cannot provide default inbound access control. VAX/VMS nodes are all capable of providing default inbound access control (in addition to proxy access) by setting the NONPRIV USER, PASSWORD, and ACCOUNT parameters in the EXECUTOR database.

If proxy access is implicitly set for a node to either OUTGOING or BOTH via the EXECUTOR DEFAULT PROXY parameter, the USER, PASSWORD, and ACCOUNT parameters may still be setup for that node. In this case, outbound proxy access to that node will be inhibited since the DECnet-VAX connect message will contain non-null access control.

The OBJECT database contains the PROXY parameter to control proxy access to and from individual objects in the network. The value for this parameter is taken from the EXECUTOR DEFAULT PROXY parameter if it has not been given an explicit value, or if a given object is not defined in the database.

### 7.6.1.4

#### Conditions for Proxy Access

For proxy access to be allowed, five conditions must be satisfied. If any of these conditions are not met, the default DECnet account is used. These five conditions are:

- 1 The EXECUTOR DEFAULT PROXY parameter for the initiating node must be either BOTH or OUTGOING.
- 2 The OBJECT PROXY parameter for the initiating node must be either BOTH or OUTGOING.
- 3 The EXECUTOR DEFAULT PROXY parameter for the destination node must be either BOTH or INCOMING.
- 4 The OBJECT PROXY parameter for the destination node must be either BOTH or INCOMING.

- 5 There must be an entry in NETUAF.DAT on the destination node for the initiating node-user pair. For example, if the account HYDRA on the destination node of CRAB will permit proxy access for user CLAW on node LOBSTER, the listing of NETUAF.DAT for node CRAB would include the line entry:

LOBSTER::CLAW HYDRA

---

### 7.6.2 Special Considerations

Proxy access is in effect a selective merging of the authorization databases of the affected systems. Therefore, the security is only as good as the security of the least secure node involved.

Although one of the strengths of proxy access is that it eliminates passwords going over the network, it is possible for a personal computer to deceive the proxy login mechanism by impersonating one of the authorized nodes. For this reason, the parameter INCOMING for proxy should be used sparingly on vital nodes. Also, you should never set up a proxy account with privileges that could damage your system. In general, timesharing nodes should not permit proxy access from stand-alone nodes.

---

## 7.7 Sharing and Exchanging Files in the Network Environment

The easiest way for a user to transfer a text file to another user is to invoke the VAX/VMS Mail Utility and send the user a copy of the file. This method is reasonably secure, since passwords need not be revealed, and the original protection of the file is not changed. The receiving user simply includes a new file name with the MAIL command EXTRACT/NOHEADER to place a copy in the user's own directory. The copy automatically acquires the user's default protection. Next, the user would typically use the MAIL command DELETE to remove the copy from the mail file.

These steps are simple enough for single transfers of text files between a few users. However, they are inappropriate for nontext files such as binary files. Furthermore, different procedures become attractive once greater numbers of files and users become involved.

Sites that take security seriously should discourage users from any temptation to share passwords or change their file and/or directory protection codes to grant the WORLD category READ or EXECUTE access. Granting the BYPASS or READALL privileges wantonly is even less desirable. The only really secure method for sharing and exchanging files in the network environment is to set up proxy accounts and place ACLs on the directories and files.

---

### 7.7.1 Many Remote Users Seek Access for a Single Purpose

It is not unusual for a network manager to face the need to admit a number of users from outside nodes into a directory on the home node for a particular purpose. In this situation, the security manager creates a proxy account and adds the proxy records to admit the outsiders into that one account. Since an outsider can only hold one proxy account on the home node, this account should represent the user's only reason for accessing the node. Also note that there may be a number of users on the home node who need to share the files in this account's directory, as well. To provide that access while protecting the files from outsiders, ACLs can be placed on the directory and files.

Consider an example where a central depository is needed for sales update information that a number of users scattered throughout the corporation need to read. The security manager at the node (BERTHA) where the files will reside, creates the special account SALES\_READER. SALES\_READER is set up as a captive account with mail disabled. The default directory is [SALESINFO], which has the following default protection code:

```
(S:RWED,O:RWED,G:R,W)
```

Note that this protection code permits users in the same group as SALES\_READER on the home node BERTHA to read the files. Furthermore, only the users in the system category, the owner category, or those enjoying privileges that give them such access, can update the files in the directory. ACLs are used to further define the access, as shown later below.

Next, the security manager uses the AUTHORIZE command ADD/PROXY to add the proxy records for the outside users. For example, to extend proxy access to user JACKSON on node DEXTER and user GOODWIN on node BANGOR, the commands would be as follows:

```
ADD/PROXY DEXTER::JACKSON SALES_READER
ADD/PROXY BANGOR::GOODWIN SALES_READER
```

If a little later it becomes clear that other users at the home node BERTHA need access and they do not belong to the same group as SALES\_READER, ACLs could be added to the files in the directory [SALESINFO]. For example, suppose R\_GRANT needs CONTROL access to all the files and J\_MAGOON needs READ access to all the files. The following two DCL commands would define the ACL for the directory and then propagate it to all the existing files:

```
$ SET ACL/ACL=-
$_((IDENTIFIER=R_GRANT,OPTIONS=DEFAULT,ACCESS=CONTROL),-
$_(IDENTIFIER=J_MAGOON,OPTIONS=DEFAULT,ACCESS=READ))-
$_[000000]SALESINFO.DIR
$ SET FILE/ACL/DEFAULT *.*;*
```

### 7.7.2

#### Remote Users from One Node Require Single Account Access

It is possible to encounter situations where all (or nearly all) the users at a remote node require access to one of your accounts. For example, if all the users at node CARIBOU require access to the account SALES\_READER, you could specify the following ADD/PROXY command:

```
ADD/PROXY CARIBOU::* SALES_READER
```

Before you do this, be absolutely certain that you really want to admit every user from the remote node. Check to be certain that there are no guest accounts or other undesired accounts at the remote node. If you discover there are a few exceptions at the remote node, you cannot simply remove the extra users with REMOVE/PROXY commands because the ADD/PROXY command above creates a single entry in NETUAF.DAT. However, you might possibly use the following technique to exclude selected entries:

```
ADD/PROXY CARIBOU::* SALES_READER
ADD/PROXY CARIBOU::FRED DECNET
ADD/PROXY CARIBOU::GEORGE DECNET
```

In the example above, all users on node CARIBOU use the SALES\_READER proxy account except users FRED and GEORGE, who are directed to use DECNET.

### 7.7.3

#### A Few Outside Users Require Access for Multiple Purposes

The techniques outlined above work well when there are a fair number of outside users requiring access for one specific purpose. However, you may also encounter situations where smaller numbers of outsiders need access for multiple purposes, involving files that require special protection. In this case you would want to consider setting up individual accounts for the users, and you would apply ACLs more extensively.

For example, a large corporation with many branch offices might find it desirable to establish several proxy accounts for specific purposes for sharing. Assume the central corporation offices want to grant two key users from their two East Coast nodes READ and WRITE access to the project files for code name LEVIGRAY and READ only access to the BETSEYHARLOW project files. At the same time, there are three users from the West Coast who need READ access to those LEVIGRAY files and require READ and WRITE access to the BETSEYHARLOW files. Only two users from the central office will have full access rights to the LEVIGRAY files and two other users from headquarters will have full access rights to the BETSEYHARLOW files. For working purposes, the situation could be represented in tabular form as shown in Figure 7-7.

**Figure 7-7 Example of Protected File Sharing in a Network**


---

Access Requirements to CENTRL::PROJ:[DESGN_PROJECTS] Owned by [DESIGNERS,MGR]		
Users & Nodes	Subdirectory LEVI Project Files LEVIGRAY*. *	Subdirectory BETSEY Project Files BETSEYHARLOW*. *
FRISCO::ALBION	R	RW
FRISCO::ELTON	R	RW
LA::IRVING	R	RW
CENTRL::DIANTHA	RWED	NONE
CENTRL::BRITTANIA	RWED	NONE
CENTRL::ALBERT	NONE	RWED
CENTRL::DELIA	NONE	RWED
BOS::AYLMER	RW	R
WASH::LAVINA	RW	R

---

The following solution uses five proxy accounts in addition to the four local accounts on CENTRL, plus ACLs on the directory, subdirectories, and files. First, the security manager at headquarters uses AUTHORIZE to create new accounts on node CENTRL, for the remote users ALBION, ELTON, IRVING, AYLMEER, and LAVINA. These accounts should be captive, disallow mail, and be restricted to network access only. The accounts are even restricted to a subset of DCL through command language tables. The default directory should be [DESGN\_PROJECTS] for each user. The manager decides it makes sense to put them into the DESIGNER group to match their proposed uses of the files.

Presumably accounts already exist for DIANTHA, BRITTANIA, ALBERT, and DELIA. They need not necessarily belong to the same group. They will be informed which device and directory to use for their work.

The next step is to add the proxy records to the NETUAF, with the following AUTHORIZE commands:

```
ADD/PROXY FRISCO::ALBION ALBION
ADD/PROXY FRISCO::ELTON ELTON
ADD/PROXY LA::IRVING IRVING
ADD/PROXY BOS::AYLMER AYLMEER
ADD/PROXY WASH::LAVINA LAVINIA
```

## Security for a DECnet Node

The security manager at node CENTRL places an ACL on the top-level directory for [DESGN\_PROJECTS] with the following DCL command:

```
$ SET ACL/ACL=(DEFAULT_PROTECTION,S:RWED,O,G,W) -  
$_[000000]DESGN_PROJECTS.DIR
```

This ensures that no one outside of the SYSTEM category of users can gain any UIC-based access to the files in the directory or any of the subdirectories, unless they possess the BYPASS privilege. In fact, this restriction applies to those five users in the group DESIGNERS, as well. The plan is for all files to possess ACLs that will admit the select group of users. It is desirable to propagate this protection code to all the files in this directory and its subdirectories. (Remember, the ACLs that will be placed on the files for further protection will take precedence when one of these users actually seeks access to a file.)

Two subdirectories are created in [DESGN\_PROJECTS]:

- [DESGN\_PROJECTS.LEVI]
- [DESGN\_PROJECTS.BETSEY]

Next, the security manager uses the VAX/VMS ACL Editor to place the following additional ACEs in the ACL for the top-level directory:

```
DESGN_PROJECTS.DIR  
(IDENTIFIER=DIANTHA,OPTIONS=PROTECTED,ACCESS=EXECUTE)  
(IDENTIFIER=BRITTANIA,OPTIONS=PROTECTED,ACCESS=EXECUTE)  
(IDENTIFIER=ALBERT,OPTIONS=PROTECTED,ACCESS=EXECUTE)  
(IDENTIFIER=DELIA,OPTIONS=PROTECTED,ACCESS=EXECUTE)  
(IDENTIFIER=AYLMER,OPTIONS=PROTECTED,ACCESS=EXECUTE)  
(IDENTIFIER=LAVINA,OPTIONS=PROTECTED,ACCESS=EXECUTE)  
(IDENTIFIER=ALBION,OPTIONS=PROTECTED,ACCESS=EXECUTE)  
(IDENTIFIER=ELTON,OPTIONS=PROTECTED,ACCESS=EXECUTE)  
(IDENTIFIER=IRVING,OPTIONS=PROTECTED,ACCESS=EXECUTE)
```

These protected ACEs ensure that only the select nine users can access the top-level directory. Since no one receives WRITE or DELETE access to the top directory through the ACL, the directory and subdirectories are generally protected from deletion and renaming of files. (Of course, the SYSTEM category of user obtains WRITE and DELETE access through the UIC-based protection.)

## Security for a DECnet Node

Next, the security manager must create ACLs on the subdirectories. The ACEs that are required are shown for their respective subdirectories:

[DESGN\_PROJECTS]LEVI.DIR

```
(IDENTIFIER=DIANTHA,OPTIONS=PROTECTED,ACCESS=READ+WRITE+EXECUTE+CONTROL)
(IDENTIFIER=DIANTHA,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ+WRITE+EXECUTE+DELETE+CONTROL)
(IDENTIFIER=BRITTANIA,OPTIONS=PROTECTED,ACCESS=READ+WRITE+EXECUTE+CONTROL)
(IDENTIFIER=BRITTANIA,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ+WRITE+EXECUTE+DELETE+CONTROL)
(IDENTIFIER=AYLMER,OPTIONS=PROTECTED,ACCESS=READ+WRITE)
(IDENTIFIER=AYLMER,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ+WRITE)
(IDENTIFIER=LAVINA,OPTIONS=PROTECTED,ACCESS=READ+WRITE)
(IDENTIFIER=LAVINA,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ+WRITE)
(IDENTIFIER=ALBION,OPTIONS=PROTECTED,ACCESS=READ)
(IDENTIFIER=ALBION,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ)
(IDENTIFIER=ELTON,OPTIONS=PROTECTED,ACCESS=READ)
(IDENTIFIER=ELTON,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ)
(IDENTIFIER=IRVING,OPTIONS=PROTECTED,ACCESS=READ)
(IDENTIFIER=IRVING,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ)
```

[DESGN\_PROJECTS]BETSEY.DIR

```
(IDENTIFIER=ALBERT,OPTIONS=PROTECTED,ACCESS=READ+WRITE+EXECUTE+CONTROL)
(IDENTIFIER=ALBERT,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ+WRITE+EXECUTE+DELETE+CONTROL)
(IDENTIFIER=DELIA,OPTIONS=PROTECTED,ACCESS=READ+WRITE+EXECUTE+CONTROL)
(IDENTIFIER=DELIA,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ+WRITE+EXECUTE+DELETE+CONTROL)
(IDENTIFIER=ALBION,OPTIONS=PROTECTED,ACCESS=READ+WRITE)
(IDENTIFIER=ALBION,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ+WRITE)
(IDENTIFIER=ELTON,OPTIONS=PROTECTED,ACCESS=READ+WRITE)
(IDENTIFIER=ELTON,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ+WRITE)
(IDENTIFIER=IRVING,OPTIONS=PROTECTED,ACCESS=READ+WRITE)
(IDENTIFIER=IRVING,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ+WRITE)
(IDENTIFIER=AYLMER,OPTIONS=PROTECTED,ACCESS=READ)
(IDENTIFIER=AYLMER,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ)
(IDENTIFIER=LAVINA,OPTIONS=PROTECTED,ACCESS=READ)
(IDENTIFIER=LAVINA,OPTIONS=DEFAULT+PROTECTED,ACCESS=READ)
```

You will note that both of the ACLs above include two ACEs for each identifier. The first ACE controls the access to the subdirectory. It denies delete access for the protection of the subdirectory and is not propagated to all the files created in the subdirectory. The second ACE for each identifier will automatically propagate to all files added to their respective subdirectories because of the inclusion of the OPTIONS=DEFAULT option specification. Furthermore, the option PROTECTED ensures that all the ACEs are protected from deletion except by specific action. At this point all the ground work has been completed. Over time files are added to the subdirectories. Thus, when the user LAVINA in Washington issues the DCL command below, the file LEVIGRAYMEM3.MEM is printed at node WASH:

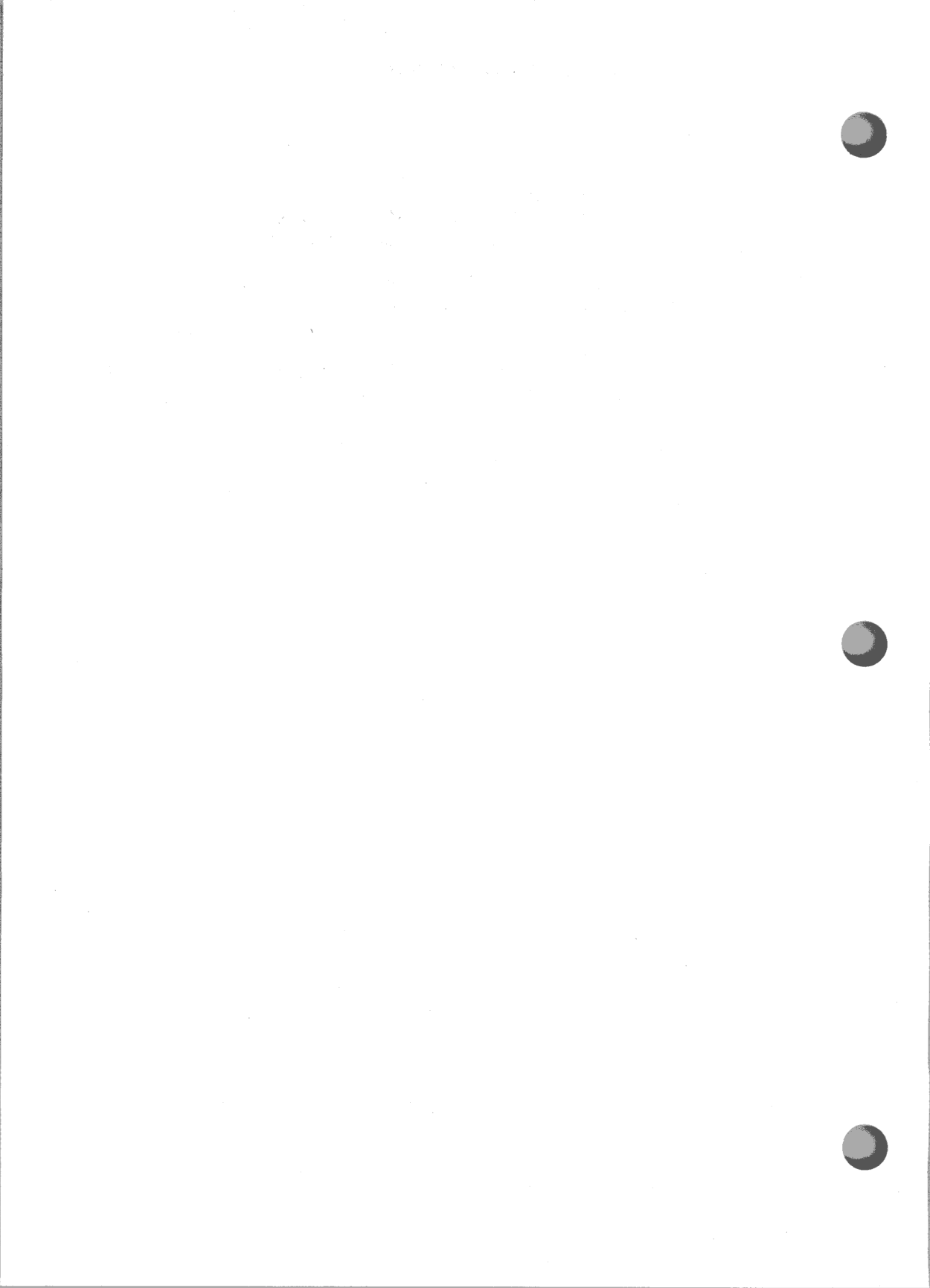
```
$ COPY CENTRL::LEVIGRAYMEM3.MEM LP:
```



## Security for a DECnet Node

However, any attempts by user LAVINA to edit the file BETSEYHARLOWMEM8.MEM would fail, because user LAVINA is denied WRITE access through the ACL.

If there were a greater number of users involved in this scheme, it would soon become worthwhile to grant additional identifiers to the users. For example, each user who would be allowed READ access to the files in the LEVI subdirectory might be given the identifier LEVL\_READER, and so forth. The ACLs could then be shortened somewhat.



# 8

## Security Concerns on a Cluster

---

This chapter describes topics of concern to security managers on clustered VAX/VMS systems. All readers of this chapter should have previously read and mastered the information in the *Guide to VAXclusters*.

The *Guide to VAXclusters* describes the person or team of persons who manage a VAXcluster as the cluster manager. The cluster manager, of course, is a specialized system manager. Just where the security manager role fits into this picture is highly site-dependent. At some VAXcluster sites, the security manager role will be performed by the cluster manager. At other sites there may be one or more security managers in addition to a cluster management team.

When a site chooses to separate the security manager job, the security management function remains as important as ever and demands especially good coordination, cooperation, and communication. As in previous chapters, this chapter will use the title of security manager to refer to individuals who have the responsibility for system security, regardless of what other responsibilities they may or may not hold. As you might expect, the security manager for a VAXcluster will generally require the training and skills of the cluster manager.

---

### 8.1 Overview of Clusters and Security Considerations

To review briefly, *clustered VAX/VMS systems* refer to those systems that use the VAX/VMS hardware and software that permits sharing of disks, resources, and even a common operating system among various VAX-11 computers. The computers are said to be joined in a VAXcluster. Clustered computers are connected together through a special high speed hardware computer interconnect known as the CI. Furthermore, there are two types of VAXclusters: homogeneous and nonhomogeneous (or heterogeneous). A *homogeneous*

VAXcluster refers to one in which the operating system environment is identical on each member node, whereas on a *nonhomogeneous* VAXcluster each node has a unique environment.

From a security standpoint, the fact that a node is part of a VAXcluster generally has little significance. This is because each node in a VAXcluster appears to operate as a single system. Thus, all the security features previously described apply equally well to any node in the cluster. The only difference is that when a security manager implements the feature on one node of a homogeneous cluster, all the nodes are affected. The reason for this is that each cluster node mediates access by its subjects to all objects in the cluster. In effect, the cluster operates within a single security perimeter, with the reference monitor on each node acting as a gateway through that perimeter.

There is however, one area where the actions the cluster manager takes in setting up the VAXcluster can affect the security operations of the system. This area concerns the creation and management of various pieces of the overall authorization database. This chapter elaborates on those security implications and provides recommendations.

---

### 8.2 Authorization Database Considerations

On a VAXcluster there are advantages when all elements of the user authorization data exist in a common database. These authorization elements include the system and network user authorization files and the rights database, which are present on all VAX/VMS systems, and the optional autologin file, SYSALF.DAT, which some sites will create with the procedure SYS\$MANAGER:ALFMAINT.COM. (Section 5.2.9 describes how to use ALFMAINT.COM.) If you decide to create an autologin file, consider maintaining it in a common authorization database along with your authorization files and rights database. Also, remember that on a clustered system the autologin file must include the cluster node name as a prefix to the terminal name. For example, the terminal TTA0 on node WILLOW would be represented as WILLOW\$TTA0.

## Security Concerns on a Cluster

The reasons for maintaining your authorization elements on a common disk are as follows:

- Centralized management of the data is facilitated, which in turn saves time and errors.
- The system requires a certain amount of consistency, anyway. That is, if you want any of the three primary elements (NETUAF, SYSUAF, or RIGHTSLIST) to be common for all nodes, you must have all three common. This requirement reflects the fact that AUTHORIZE performs some automatic maintenance functions on the data.

You are not absolutely required to set up a common shared disk for these files, but you are encouraged to do so. If you choose to ignore this recommendation, remember that coordination is required in your choices for UICs, group numbers and identifiers. Make certain they are all unique for unique purposes. Also, each user should have the same UIC, group number, and set of identifiers defined on every node.

---

### 8.3 Building a Common User Environment

The *Guide to VAXclusters* provides detailed guidelines for building a common user environment. The procedures depend on the initial state of your system. As a result, you must study the procedures in the proper context and apply them according to the guidelines that you will find in that document. For this reason, no attempt is made to reproduce the procedures here.

---

### 8.4 File Sharing Considerations

It is important to realize that when disks are shared, the file system works locally on each node to perform the file protection checking. By setting up separate authorization files for each node, you could fail to recognize the actual privileges and access that users have. For example, suppose user RBIRCH is authorized for the SETPRV privilege on node ELM but not on node MAPLE and is a member of the BIOSTAFF group on node ELM but not on node MAPLE. The security manager on MAPLE might initially conclude that files on the disk BOTANYDISK that admitted only users in the BIOSTAFF group were protected from RBIRCH. However, since BOTANYDISK is a shared disk, and RBIRCH has that account on ELM that

gives access, RBIRCH can always gain access to the files by logging into the ELM node, even when unable to gain the same access through the MAPLE node. The disk is no more protected than it is at its least protected node! This situation argues in favor of uniform authorization of users, so that the protection is straightforward and such deceptions are unlikely to occur.

---

### 8.5 Using DECnet Between Cluster Nodes

While VAXclusters offer special communication facilities for the most common operations (such as file sharing, lock management, and so forth), other VAX/VMS features may require the use of DECnet to be used across a cluster. Examples of such intra-cluster use would include the need to access disks that are not cluster-accessible and higher-level features available through DCL commands such as SHOW USERS, and so forth.

For maximum coherence in DECnet operations, you should set up a proxy database that maps users into their own accounts when they initiate DECnet operations. Thus, for each node in a homogeneous cluster, you would add a proxy file record using the following AUTHORIZE command:

```
ADD/PROXY node:: * *
```

However, if you are running a nonhomogeneous cluster, you will need a more complex arrangement of proxies to only cross-map your users as they are authorized.

---

### 8.6 Summary

In summary, security operations are enhanced on a VAXcluster when all the authorization data resides on a common shared disk. The files of concern are:

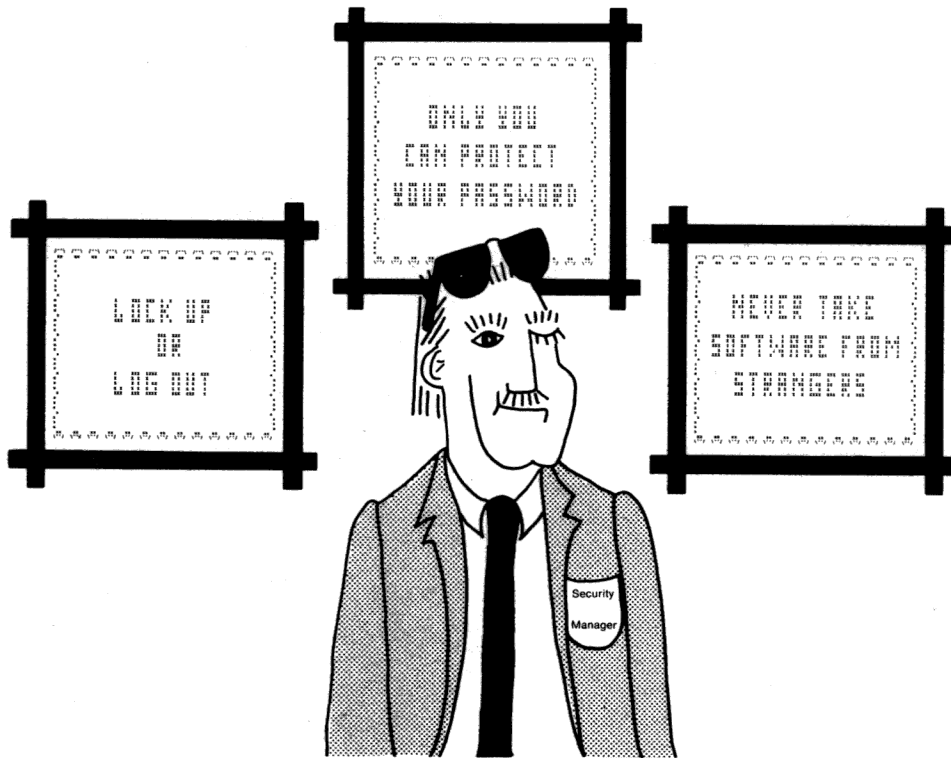
- SYSUAF.DAT
- NETUAF.DAT
- RIGHTSLIST.DAT
- SYSALF.DAT (if it exists)

Each user must have the same UIC, group number and set of identifiers defined on each cluster node.

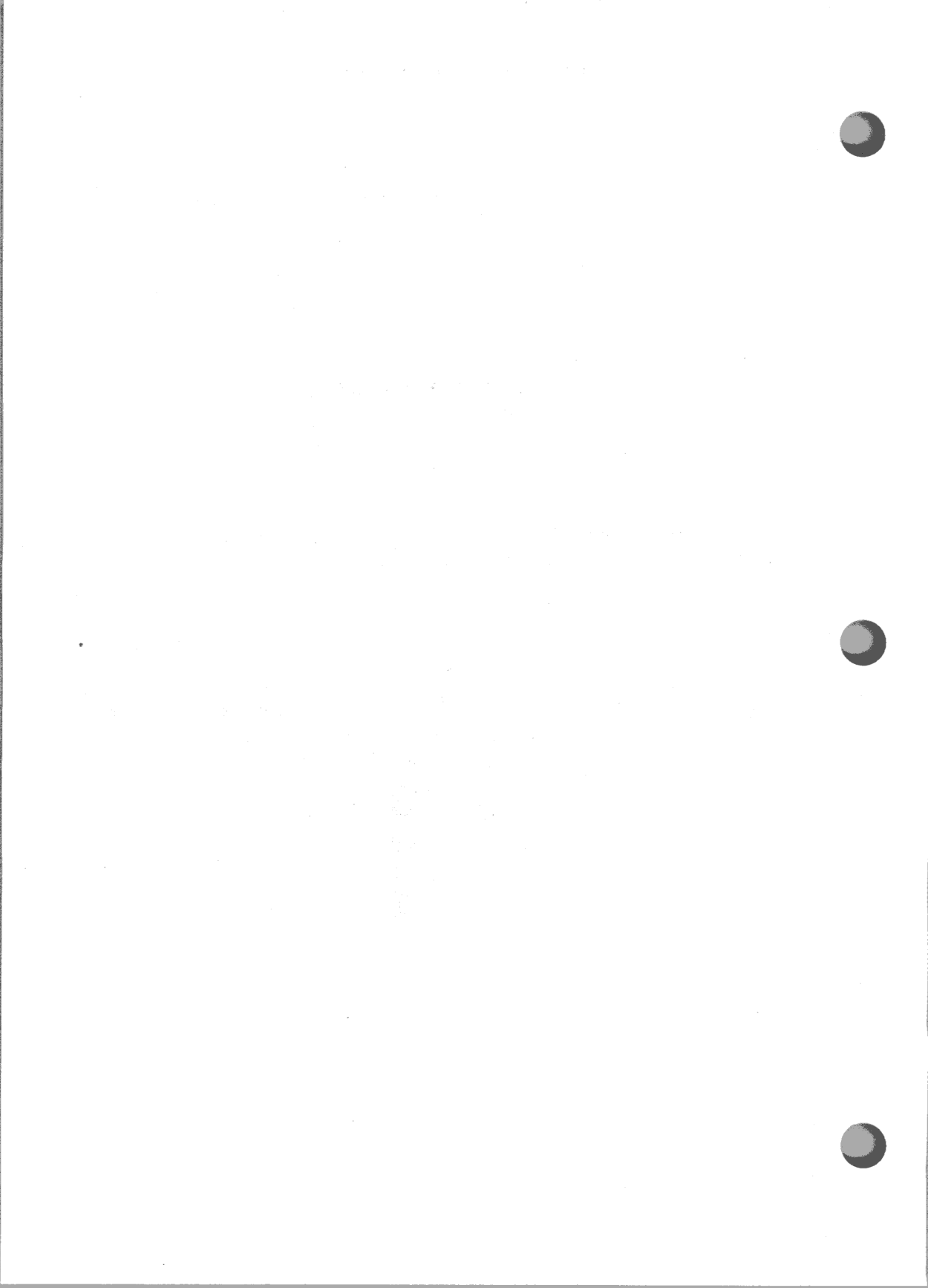
## Security Concerns on a Cluster

On a shared disk, the protection of a file from a specific user cannot effectively exceed the maximum access that user can gain from one of the nodes.

In all respects, the VAX/VMS security features operate the same on clustered systems as they do on nonclustered systems.



ZK-2067-84





# A

---

## Privileges

---

### A.1 User Privileges

This appendix describes all the privileges available on VAX/VMS systems, including the abilities passed by each privilege and the appropriate users to receive them.

---

#### A.1.1 ACNT Privilege

Only a user who has the ACNT privilege can create subprocesses or detached processes in which accounting is disabled. Thus, only such a privileged user can issue the DCL command RUN with the /NOACCOUNTING qualifier or inhibit accounting in the Create Process (\$CREPRC) system service.

---

#### A.1.2 ALLSPOOL Privilege

The ALLSPOOL privilege allows the user's process to allocate a spooled device by executing the Allocate Device (\$ALLOC) system service or by using the DCL command ALLOCATE.

The \$ALLOC system service lets a process allocate, or reserve, a device for its exclusive use. A shareable mounted device cannot be allocated.

You should grant this privilege only to users who need to perform logical or physical I/O operations to a spooled device. Ordinarily, the privilege of allocating a spooled device is granted only to symbionts.

---

### A.1.3 ALTPRI Privilege

The ALTPRI privilege allows the user's process to:

- Increase its own base priority
- Set the base priority of another process to a value higher than that of the target process

The base priority is increased by executing the Set Priority (\$SETPRI) system service or the DCL command SET PROCESS /PRIORITY. As a rule, this system service lets a process set its own base priority or the base priority of another process. However, one process can only set the priority of a second process if one of the following conditions applies:

- The process calling the \$SETPRI system service has the same UIC as the target process
- The calling process has process control privilege (GROUP or WORLD) over the target process

With ALTPRI, a process can create a process with a priority higher than its own. It creates such a process by using an optional argument to the Create Process (\$CREPRC) system service or to the DCL command RUN.

You should not grant this privilege widely; if unqualified users have the unrestricted ability to set base priorities, the fair and orderly scheduling of processes for execution can easily be disrupted.

---

### A.1.4 BUGCHK Privilege

The use of this privilege should be restricted to system software supplied by DIGITAL that uses the VAX/VMS Bugcheck Facility. This privilege allows the process to make bugcheck error log entries.

---

### A.1.5 BYPASS Privilege

The BYPASS privilege allows the user's process read, write, execute, and delete access to all files, bypassing UIC-based and ACL protection.

You should grant this privilege with extreme caution, as it overrides all file protection. It should be reserved for use by either well tested, reliable programs and command procedures or the system backup operation (see the *Guide to VAX/VMS System Management and Daily Operations* and the *Guide to VAX/VMS Disk and Magnetic Tape Operations* for a discussion of backup operations). SYSPRV (see below) is adequate for interactive use, as it ultimately grants access to all files, while still providing access checks.

---

### A.1.6 CMEXEC Privilege

The CMEXEC privilege allows the user's process to execute the Change Mode to Executive (\$CMEXEC) system service.

This system service lets a process change its access mode to executive, execute a specified routine, and then return to the access mode that was in effect before the system service was called. While in executive mode, the process is allowed to execute the Change Mode to Kernel (\$CMKRNL) system service.

You should grant this privilege only to users who need to gain access to protected and sensitive data structures and internal functions of the operating system. If unqualified users have unrestricted access to sensitive data structures and functions, the operating system and service to other users can easily be disrupted. Such disruptions can include failure of the system, destruction of the database, and exposure of confidential information to unauthorized persons.

---

### **A.1.7 CMKRNL Privilege**

The CMKRNL privilege allows the user's process to execute the Change Mode to Kernel (\$CMKRNL) system service.

This system service lets a process change its access mode to kernel, execute a specified routine, and then return to the access mode that was in effect before the system service was called.

You should grant this privilege only to users who need to execute privileged instructions or who need to gain access to the most protected and sensitive data structures and functions of the operating system. If unqualified users have unrestricted use of privileged instructions and unrestricted access to sensitive data structures and functions, the operating system and service to other users can easily be disrupted. Such disruptions can include failure of the system, destruction of the database, and exposure of confidential information to unauthorized persons.

---

### **A.1.8 DETACH Privilege**

Users can create detached processes that have their own UIC without this privilege, provided the users do not exceed their MAXJOBS and MAXDETACH quotas. However, the DETACH privilege becomes valuable when a user wants to specify a different UIC for the detached process. There is no restriction on the UIC that can be specified for a detached process if you have the DETACH privilege. Thus, there are no restrictions on the files and directories to which a detached process can gain access.

The DETACH privilege allows the user's process to create detached processes by executing the Create Process (\$CREPRC) system service. Detached processes remain in existence even after the user who created them has logged off the system.

An example of a detached process is the process created by the system for a user when the user logs in to the system.

---

### **A.1.9 DIAGNOSE Privilege**

The DIAGNOSE privilege allows the user to run online diagnostic programs and to intercept and copy all messages that are written to the error log file.

---

**A.1.10 EXQUOTA Privilege**

The EXQUOTA privilege allows the space taken by the user's files on given disk volumes to exceed any usage quotas set for the user (as determined by UIC) on those volumes.

---

**A.1.11 GROUP Privilege**

The GROUP privilege allows the user's process to affect other processes in its own group by executing the following process control system services: Suspend Process (\$SUSPND), Resume Process (\$RESUME), Delete Process (\$DELPRI), Set Priority (\$SETPRI), Wake (\$WAKE), Schedule Wakeup (\$SCHDWK), Cancel Wakeup (\$CANWAK), and Force Exit (\$FORCEX). The user's process is also allowed to examine other processes in its own group by executing the Get Job/Process Information (\$GETJPI) system service. The user with the GROUP privilege can issue the following DCL commands for other processes in its group: SET QUEUE, DELETE/ENTRY, STOP/ENTRY, and SET PROCESS.

The GROUP privilege is not needed for a process to exercise control over, or to examine, subprocesses that it created or other detached processes of its UIC. You should, however, grant this privilege to users who need to exercise control over each other's processes and operations.

---

**A.1.12 GRPNAM Privilege**

The GRPNAM privilege allows the user's process to insert names into the logical name table of the group to which the process belongs and to delete names from that table by the use of the following logical name system services: Create Logical Name (\$CRELNM) and Delete Logical Name (\$DELLNM).

In addition, the privileged user can use the DCL commands ASSIGN and DEFINE to add names to the group logical name table, the DCL command DEASSIGN to delete names from the table, and the /GROUP qualifier of the DCL command MOUNT to share volumes among group members.

## Privileges

This privilege should not be granted to all users of the system because it allows the user to create an unlimited number of group logical names. When unqualified users have the unrestricted ability to create group logical names, excessive use of system dynamic memory can degrade system performance. In addition, a user with the GRPNAM privilege can interfere with the activities of other users in the same group by creating definitions of commonly used logical names such as SYS\$SYSTEM.

---

### A.1.13 GRPPRV Privilege

This privilege allows a process access to a file using the file's SYSTEM protection field when the process's group matches the group of the file owner. GRPPRV also allows a process to change the protection of any file whose owner group matches the process's group. This privilege also allows a process to change the ownership of objects within the process's group.

You should grant this privilege only to users who function as group managers. Note that if any member of a group holds any of the privileges in the "all" category, then any other member of that group who holds the GRPPRV privilege can gain control of the system, by indirectly acquiring that privilege. (The privileges in the "all" category have the potential to control the system and are described in Section 5.3.6.

---

### A.1.14 LOG\_IO Privilege

The LOG\_IO privilege allows the user's process to execute the Queue I/O Request (\$QIO) system service to perform logical-level I/O operations. LOG\_IO privilege is also required for certain device control functions, such as setting permanent terminal characteristics.

Usually, user I/O requests are handled indirectly by use of an I/O package such as VAX Record Management Services. However, to increase their control over I/O operations and to improve the efficiency of I/O operations, skilled users sometimes prefer to handle directly the interface between their process and a system I/O driver program. They can do this by executing the Queue I/O Request system service; in many instances, the operation called for is a logical-level I/O operation. Note that logical level functions are permitted

## **Privileges**

without LOG\_IO privilege on a device mounted with the /FOREIGN qualifier and on non-file-structured devices.

You should grant this privilege only to users who need it because it allows a process to access data anywhere on the selected volume without the benefit of any file structuring. If this privilege is given to unqualified users who have no need for it, the operating system and service to other users can easily be disrupted. Such disruptions can include the destruction of information on the system device, the destruction of user data, and the exposure of confidential information to unauthorized persons.

---

### **A.1.15 MOUNT Privilege**

The MOUNT privilege allows the user's process to execute the mount volume QIO function. The use of this function should be restricted to system software supplied by DIGITAL.

---

### **A.1.16 NETMBX Privilege**

The NETMBX privilege allows the user to perform functions related to a DECnet computer network. You should normally grant this privilege to general users.

---

### **A.1.17 OPER Privilege**

The OPER privilege allows the user to use the Operator Communication Manager (OPCOM) process, as follows: to reply to user's requests, to broadcast messages to all terminals logged in, to designate terminals as operators' terminals and specify the types of messages to be displayed on these operators' terminals, and to initialize and control the log file of operators' messages. In addition, this privilege lets the user set devices spooled, create and control both batch queues and print queues.

You should grant this privilege only to special users—the operators of the system. These are the users who respond to the requests of ordinary users, who tend to the needs of the system's peripheral devices (mounting reels of tape and changing printer forms), and who attend to all the other day-to-day chores of system operation. (A nonprivileged user can log

in on the console terminal to respond to operator requests, for example, to mount a tape.)

---

### **A.1.18 PFNMAP Privilege**

The PFNMAP privilege allows the user's process to map to specific pages of physical memory or I/O device registers, no matter who is using the pages or registers.

You should exercise caution in granting this privilege. If unqualified users have unrestricted access to physical memory, the operating system and service to other users can easily be disrupted. Such disruptions can include failure of the system, destruction of the database, and exposure of confidential information to unauthorized persons.

---

### **A.1.19 PHY\_IO Privilege**

The PHY\_IO privilege allows the user's process to execute the Queue I/O Request (\$QIO) system service to perform physical-level I/O operations.

Usually, users' I/O requests are handled indirectly by use of an I/O package such as VAX Record Management Services. However, to increase their control over I/O operations and to improve the efficiency of their applications, skilled users sometimes prefer to handle directly the interface between their process and a system I/O driver program. They can do this by executing the Queue I/O Request system service; in many instances, the operation called for is a physical-level I/O operation.

You should grant the PHY\_IO privilege only to users who need it; in fact, this privilege should be granted even more carefully than the LOG\_IO privilege. If this privilege is given to unqualified users who have no need for it, the operating system and service to other users can easily be disrupted. Such disruptions can include the destruction of information on the system device, the destruction of user data, and the exposure of confidential information to unauthorized persons.



---

**A.1.20 PRMCEB Privilege**

The PRMCEB privilege allows the user's process to create or delete a permanent common event flag cluster by executing the Associate Common Event Flag Cluster (\$ASCEFC) or Delete Common Event Flag Cluster (\$DLCEFC) system service. Common event flag clusters enable cooperating processes to communicate with each other and thus provide the means of synchronizing their execution.

This privilege should not be granted to all users of the system, because it allows the user to create an unlimited number of permanent common event flag clusters. A permanent cluster remains in the system even after the creating process has been terminated and continues to use up a portion of system dynamic memory. When many users have the unrestricted ability to create permanent common event flag clusters, the excessive use of system dynamic memory can degrade system performance.

---

**A.1.21 PRMGBL Privilege**

The PRMGBL privilege allows the user's process to create permanent global sections by executing the Create and Map Section (\$CRMPSC) system service. In addition, the user with this privilege (plus the CMKRNL and SYSGBL privileges) can use the VAX/VMS Install Utility.

Global sections are shared structures that can be mapped simultaneously in the virtual address space of many processes. All processes see the same code or data. Global sections are used for reentrant subroutines or data buffers.

You should grant this privilege with care. If permanent global sections are not explicitly deleted, they tie up space in the global section and global page tables, which are limited resources.

---

### **A.1.22 PRMMBX Privilege**

The PRMMBX privilege allows the user's process to create or delete a permanent mailbox by executing the Create Mailbox and Assign Channel (\$CREMBX) system service or the Delete Mailbox (\$DELMBX) system service.

Mailboxes are buffers in virtual memory that are treated as if they were record-oriented I/O devices. A mailbox is used for general interprocess communication.

The PRMMBX privilege should not be granted to all users of the system. Permanent mailboxes are not automatically deleted when the creating processes are deleted and, thus, continue to use up a portion of system dynamic memory.

---

### **A.1.23 PSWAPM Privilege**

The PSWAPM privilege allows the user's process to control whether it can be swapped out of the balance set by executing the Set Process Swap Mode (\$SETSWM) system service. Not only must a process have this privilege to lock itself in the balance set (that is, to disable swapping), but also to unlock itself (that is, to enable swapping).

With this privilege, a process can create a process that is locked in the balance set (process swap mode disabled) by using an optional argument to the Create Process (\$CREPRC) system service or, when the DCL command RUN is used to create a process, by using a qualifier of the RUN command.

You should grant this privilege only to users who need to lock a process in memory for performance reasons. Typically, this will be a real-time process. If unqualified users have the unrestricted ability to lock processes in the balance set, physical memory can be held unnecessarily and thereby degrade system performance.

---

### A.1.24 READALL Privilege

The READALL privilege allows the process to bypass existing restrictions that would otherwise prevent the process from reading a file. However, unlike the BYPASS privilege, which permits writing and deleting, READALL only permits reading of the file and control operations (such as changing protection and writing the backup date).

You should grant this privilege to operators so that they can perform system backups. The implications of this privilege are the same as those for the SYSPRV privilege.

---

### A.1.25 SECURITY Privilege

The SECURITY privilege allows a process to perform security related functions such as enabling or disabling security audits or setting the system password.

You should grant this privilege only to security managers. Irresponsible users who obtain this privilege can subvert the system's security auditing and can lock out users through improper application of system passwords.

---

### A.1.26 SETPRV Privilege

The SETPRV privilege allows the user's process to create processes whose privileges are greater than its own, by executing the Create Process (\$CREPRC) system service with an optional argument, or by issuing the DCL command RUN to create a process. A user with this privilege can also execute the DCL command SET PROCESS/PRIVILEGES to obtain any desired privilege.

You should exercise the same caution in granting the SETPRV privilege as in granting any other privilege, since SETPRV allows the user to enable any or all privileges.

---

**A.1.27 SHARE Privilege**

The SHARE privilege allows the user's process to assign channels to devices allocated to other users.

Normally you should only grant this privilege to system software such as print symbionts. This privilege would allow an irresponsible user to interfere with the operation of devices belonging to other users.

---

**A.1.28 SHMEM Privilege**

The SHMEM privilege allows the user's process to create global sections and mailboxes (permanent and temporary) in multiport memory, if the process also has appropriate PRMGBL, PRMMBX, SYSGBL, and TMPMBX privileges. Just as in local memory, the space required for a multiport memory temporary mailbox counts against the buffered I/O byte count limit (BYTLM) of the process.

---

**A.1.29 SYSGBL Privilege**

The SYSGBL privilege allows the user's process to create system global sections by executing the Create and Map Section (\$CRMPSC) system service. In addition, the user with this privilege (plus the CMKRNL and PRMGBL privileges) can use the VAX/VMS Install Utility.

You should exercise caution in granting this privilege. System global sections require space in the global section and global page tables, which are limited resources.

---

**A.1.30 SYSLCK Privilege**

The SYSLCK privilege allows the user's process to lock system-wide resources with the Enqueue Lock Request (\$ENQ) system service. You should grant this privilege to users who need to run programs that lock resources in the system-wide resource name space.

## Privileges

You should exercise caution in granting this privilege. Users who hold the SYSLCK privilege can interfere with the synchronization of system software and all other user software as well.

---

### A.1.31 **SYSNAM Privilege**

The SYSNAM privilege allows the user's process to insert names into the system logical name table and to delete names from that table by using the Create Logical Name (\$CRELNM) and Delete Logical Name (\$DELLNM) system services. This privilege also permits the creation of executive mode logical names.

In addition, the user with this privilege can use the DCL commands ASSIGN and DEFINE to add names to the system logical name table, and can use the DEASSIGN command to delete names from the table.

You should grant this privilege only to the system operators or to system programmers who need to define system logical names (such as names for user devices, library directories, and the system directory). For example, to mount or dismount a system volume, which entails defining a system logical name, you must have the SYSNAM privilege. Note that a user with SYSNAM privilege could redefine such critical system logical names as SYS\$SYSTEM and SYSUAF, thus gaining control of the system.

---

### A.1.32 **SYSPRV Privilege**

The SYSPRV privilege allows the user to access objects by the SYSTEM protection field and to change the owner UIC and protection of a file. Even if a file is protected against system access, the user with the SYSPRV privilege can simply change the file's protection to gain access to it.

You should exercise caution in granting this privilege. Normally you would only grant this privilege to system managers and security managers. If unqualified users have system access rights, the operating system and service to others can easily be disrupted. Such disruptions can include failure of the system, destruction of the database, and exposure of confidential information to unauthorized persons.

---

### **A.1.33      TMPMBX Privilege**

The TMPMBX privilege allows the user's process to create a temporary mailbox by executing the Create Mailbox and Assign Channel (\$CREMBX) system service.

Mailboxes are buffers in virtual memory that are treated as if they were record-oriented I/O devices. A mailbox is used for general interprocess communication. Unlike a permanent mailbox, which must be explicitly deleted, a temporary mailbox is deleted automatically when it is no longer referenced by any process. Note that this privilege is required to use the DCL commands SUBMIT and PRINT.

You should usually grant this privilege to all users of the system to facilitate interprocess communication. System performance is not likely to be degraded by permitting the creation of temporary mailboxes, because their number is controlled by limits on the use of system dynamic memory (BYTLM quota).

---

### **A.1.34      VOLPRO Privilege**

The VOLPRO privilege allows the user to (1) initialize a previously used volume with an owner UIC different from the user's own UIC; (2) override the expiration date on a tape or disk volume owned by another user; (3) use the /FOREIGN qualifier to mount a Files-11 volume owned by another user; and (4) override the owner UIC protection of a volume. The VOLPRO privilege only permits control over volumes the user can mount or initialize. Volumes mounted with the /SYSTEM qualifier are safe from the user with the VOLPRO privilege as long as the user does not also have the SYSNAM privilege.

You should exercise extreme caution in granting the VOLPRO privilege. If unqualified users can override volume protection, the operating system and service to others can be disrupted. Such disruptions can include destruction of the database and exposure of confidential information to unauthorized persons.

---

**A.1.35      WORLD Privilege**

The WORLD privilege allows the user's process to affect other processes both inside and outside its group by executing the following process control system services: Suspend Process (\$SUSPND), Resume Process (\$RESUME), Delete Process (\$DELPRI), Set Priority (\$SETPRI), Wake (\$WAKE), Schedule Wakeup (\$SCHDWK), Cancel Wakeup (\$CANWAK), and Force Exit (\$FORCEX). The user's process is also allowed to examine processes outside its own group by executing the Get Job/Process Information (\$GETJPI) system service. The user with the WORLD privilege can issue the DCL commands SET QUEUE, DELETE/ENTRY, STOP/ENTRY, and SET PROCESS for all other processes.

To exercise control over subprocesses that it created or to examine these subprocesses, a process needs no special privilege. To affect or to examine other processes inside its own group, a process needs only the GROUP privilege. But to affect or examine processes outside its own group, a process needs the WORLD privilege.





# B

## Using the User Data Areas in UAF Records

Users can use VAX RMS to access UAF records for the storage and retrieval of up to 255 bytes of user data. The format of a UAF record is defined in the module \$UAFDEF in SYS\$LIBRARY:LIB.MLB. You may also find it useful to read the UAF\$ section of SYS\$LIBRARY:LIB.REQ, which contains commented structure definitions.

You can access the UAF records sequentially through the following keys:

- User name—The primary key is the user name (as specified to AUTHORIZE), a character field of size UAF\$\$\_USERNAME located at relative offset UAF\$T\_USERNAME.
- UIC—The secondary key is the UIC, located at relative offset UAF\$L\_UIC, consisting of two binary subfields of one word each. The subfields are named UAF\$W\_GRP (high-order word) and UAF\$W\_MEM (low-order word).

To place data in a UAF record, the user should take the following steps, which are designed to protect programs against future changes to the format of the UAF:

- 1 Read the UAF record.
- 2 Check the value of UAF\$W\_USRDATOFF. If it is zero, insert the current size of the record, as found in the VAX RMS record access block (RAB), into UAF\$W\_USRDATOFF. (In VAX/VMS Version 4.0, the system initializes this field to zero. Inserting the current size of the record has the effect of placing the user data at the end of the record. However, future changes to the UAF might require the system to fix the location of the user data. In this event, the system would initialize UAF\$W\_USRDATOFF to a non-zero value which the user must not change.)

- 3 Insert the user data at the relative offset pointed to by UAF\$W\_USRDATOFF. The data must take the form of a counted string, and the first byte must specify the number of bytes that follow. Furthermore, the total number of bytes must not exceed 256.
- 4 Modify the size of the record in the RAB to reflect the addition of the user data area, unless the current size of the record exceeds the end of the user data area (that is, the contents of UAF\$W\_USRDATOFF plus the length of the user data). (In Version 4.0, the user data area, when specified as outlined above, resides at the end of the record, so that modification of the RAB is essential. It is possible, however, that a future version of AUTHORIZE might fix the location of the user data area and its size (at 256 bytes), and locate additional system data after the user data. In this event, the user must not modify the size of the record.)
- 5 Update the record.

The user can now access the counted string by referring to UAF\$W\_USRDATOFF. For additional updates, the user does not modify UAF\$W\_USRDATOFF, since the rule is the user should not modify UAF\$W\_USRDATOFF if its value is not zero. However, if the record size changes, the user does modify the first byte of the counted string and the record size in the RAB, bearing in mind the considerations for future changes.

To avoid interfering with normal system operations, open the UAF to allow concurrent update; that is, specify to VAX RMS SHR=(GET,PUT,UPD,DEL). To update a record, you must lock it when you read it.

**Note:** The format of the UAF record and the way in which the system modifies it is subject to change in future versions of VAX/VMS. (For format changes, however, DIGITAL will provide a utility to update old UAFs.) Adherence to the above guidelines will minimize your reprogramming in the event of UAF record changes.

# C

## Protection for VAX/VMS System Files

The following display of protection codes and ownership corresponds to values that DIGITAL supplies for the system files following a normal installation. You should monitor these values regularly to ensure that no tampering has occurred. (The DCL commands DIRECTORY/SECURITY/OUTPUT and DIFFERENCES facilitate such checks.) This display was produced from the system manager's account with the following DCL command:

```
$ DIRECTORY/SECURITY/OUTPUT=SYSTEM_FILES.LIS SYS$SYSROOT:[*...]
```

Directory DB: [SYS0]

SYS0BI.DIR;1	[SYSTEM]	(RWE,RWE,RE,RE)
SYSERR.DIR;1	[SYSTEM]	(RWE,RWE,RE,RE)
SYSEXE.DIR;1	[SYSTEM]	(RWE,RWE,RE,RE)
SYSHLP.DIR;1	[SYSTEM]	(RWE,RWE,RE,RE)
SYSLIB.DIR;1	[SYSTEM]	(RWE,RWE,RE,RE)
SYSMAINT.DIR;1	[SYSTEM]	(RWE,RWE,RE,RE)
SYSMGR.DIR;1	[SYSTEM]	(RWE,RWE,RE,RE)
SYSMSG.DIR;1	[SYSTEM]	(RWE,RWE,RE,RE)
SYSTEST.DIR;1	[SYSTEM]	(RWE,RWE,RE,RE)
SYSUPD.DIR;1	[SYSTEM]	(RWE,RWE,RE,RE)

Total of 10 files.

Directory DB: [SYS0.SYSERR]

ERRLOG.SYS;1	[SYSTEM]	(RWED,RWED,RE,)
ERRSNAP.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)

Total of 2 files.

Directory DB: [SYS0.SYSEXE]

ACC.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ACLED.T.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ANALIMDMP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ANALYZBAD.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ANALYZOBJ.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ANALYZRMS.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
AUTHORIZE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
AUTOGEN.PAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
BACKUP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
BADBLOCK.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
BOOT58.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
BOOTBLOCK.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)

# Protection for VAX/VMS System Files

CDU.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CHECKSUM.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CLUSTRLQA.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CNDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CONFIGURE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CONINTERR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CONVERT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
COPY.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CRDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CREATE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CREATEFDL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CSP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CTDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CVDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CVTNAF.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CVTUAF.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DBDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DCL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DCLDEF.STB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DDDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DELETE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DIFF.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DIRECTORY.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DISKQUOTA.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DISMOUNT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DLDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DMDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DQDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DRDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DSRINDEX.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DSRTOC.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DTR.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DTRECV.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DTSEND.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DUDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DUMP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DXDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DYDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DZDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)

# Protection for VAX/VMS System Files

EDF.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
EDT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ENCRYPFAC.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERF.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFBRIEF.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFBUS.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFDISK.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFINICOM.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFPROC1.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFPROC2.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFPROC3.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFPROC4.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFPROC5.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFRLTIM.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFSUMM.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFTAPE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERRFMT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
EVL.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
EVL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
EXCHANGE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
F11AACP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
F11BXQP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
FAL.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
FAL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
FILESERV.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
FPEMUL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
HLD.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
HLD.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
IMGDEF.STB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
INIT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
INPSMB.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
INSTALL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
JOBCTL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LADRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LALOAD.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LALoader.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LATCP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LCDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LIBRARIAN.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LINK.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LOGINOUT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LPDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LTDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)

# Protection for VAX/VMS System Files

MACRO32.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MAIL.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MAIL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MAILEDIT.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MBXDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MESSAGE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MIRROR.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MIRROR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MODPARAMS.DAT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MOM.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MOM.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MONITOR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MP.STB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MSCP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MTAAACP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NCP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NDDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NETACP.EXE;2	[SYSTEM]	(RWED,RWED,RWED,RE)
NETCIRC.DAT;1	[SYSTEM]	(RWED,RWED,,)
NETCONF.DAT;1	[SYSTEM]	(RWED,RWED,,)
NETDEF.STB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NETDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NETLINE.DAT;1	[SYSTEM]	(RWED,RWED,,)
NETLOGING.DAT;1	[SYSTEM]	(RWED,RWED,,)
NETNODE.DAT;1	[SYSTEM]	(RWED,RWED,,)
NETOBJECT.DAT;1	[SYSTEM]	(RWED,RWED,,)
NETSERVER.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NETSERVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NETUAF.DAT;1	[SYSTEM]	(RWED,RWED,,)
NICONFIG.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NICONFIG.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NML.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NML.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NODRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NOTICE.TXT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
OPCCRASH.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
OPCOM.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
PADRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
PAGEFILE.SYS;1	[SYSTEM]	(RWED,RWED,,)
PARAMS.DAT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
PATCH.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
PHONE.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
PHONE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
PRTSMB.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
PUDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)

# Protection for VAX/VMS System Files

QUEMAN.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
RECLAIM.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
REMACP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
RENAME.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
REPLY.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
REQUEST.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
RIGHTSLIST.DAT;1	[SYSTEM]	(RWED,RWED,R,R)
RMS.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
RMS.STB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
RMSDEF.STB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
RTB.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
RTPAD.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
RTTDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
RUNDET.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
RUNOFF.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SCSDEF.STB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SCSLOA.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SDA.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SDLNPARSE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SEARCH.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SET.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SETP0.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SETPARAMS.DAT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SETSHOACL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SHOW.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SHUTDOWN.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SHWCLSTR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SMGBLDTRM.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SMGMAPTRM.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SMGTERMS.TXT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SORTMERGE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SRTRN.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
STABACCOP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
STABACKUP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
STACONFIG.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
STANDCONF.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
STARTUP.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
STARTUP.INS;3	[SYSTEM]	(RWED,RWED,RWED,RE)
STASYSGEN.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
STOPREM.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SUBMIT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SUMSLP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SWAPFILE.SYS;1	[SYSTEM]	(RWED,RWED,,)
SYS.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYS.MAP;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYS.STB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYSBOOT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYSDEF.STB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYSDUMP.DMP;1	[SYSTEM]	(RWED,RWED,,)
SYSGEN.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYSINIT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYSLOA730.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYSLOA750.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYSLOA780.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYSLOA790.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYSUAF.DAT;1	[SYSTEM]	(RWED,RWED,,)
SYSUAF.RL2;1	[SYSTEM]	(RWED,RWED,RWED,RE)

# Protection for VAX/VMS System Files

TECO.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TERMTABLE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TERMTABLE.TXT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TFDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TMDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TSDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TTDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TUDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TYPE.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UNLOCK.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
VAXVMSSYS.PAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
VERIFY.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
VMB.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
VMOUNT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
VMSHELP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
WRITEBOOT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XADRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XDDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XEDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XFDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XFLOADER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XGDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XMDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XWDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
YCDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
YFDRIVER.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)

Total of 223 files.

Directory DB: [SYS0.SYSHLP]

ACLED.T.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ANLRMSHLP.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DEBUGHLP.HLB;2	[SYSTEM]	(RWED,RWED,RWED,RE)
DISKQUOTA.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
EDFHLP.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
EDTHLP.HLB;2	[SYSTEM]	(RWED,RWED,RWED,RE)
EDTVT100.DOC;1	[SYSTEM]	(RWED,RWED,RWED,RE)
EDTVT52.DOC;1	[SYSTEM]	(RWED,RWED,RWED,RE)
EXAMPLES.DIR;1	[SYSTEM]	(RWE,RWE,RE,RE)
EXCHNGHLP.HLB;2	[SYSTEM]	(RWED,RWED,RWED,RE)
HELPLIB.HLB;2	[SYSTEM]	(RWED,RWED,RWED,RE)
INSTALHLP.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LATCP.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MAILHELP.HLB;2	[SYSTEM]	(RWED,RWED,RWED,RE)
MNRHELP.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NCPHELP.HLB;2	[SYSTEM]	(RWED,RWED,RWED,RE)
PATCHHELP.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
PHONEHELP.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SDA.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SHWCLHELP.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYSGEN.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TECO.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UAFHELP.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
VMSTLRHLP.HLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)

Total of 24 files.



# Protection for VAX/VMS System Files

Directory DB: [SYSO.SYSHLP.EXAMPLES]

ADDRIVER.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CONNECT.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DOD_ERAPAT.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DRCOPY.PRM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DRCOPYBLD.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DRMAST.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DRMASTER.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DRSLAVE.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DRSLV.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DTE_DFO3.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
GBLSECUF0.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABCHNDEF.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIO.OPT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOACQ.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOCIN.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOCIN.OPT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOCOM.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOCOMP.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOCON.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOLINK.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOPEAK.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOSAMP.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOSEC.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOSTAT.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABIOSTRT.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LABMBXDEF.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LBRDEMO.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LBRDEMO.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LBRMAC.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LPATEST.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LPMULT.B32;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MAILCOMPRESS.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MAILCVT.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MAILUAF.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MSCPMOUNT.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
PEAK.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SCRFT.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SYSGTTSTR.MSG;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TDRIVER.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TESTLABIO.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
USSDISP.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
USSLNK.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
USSTEST.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
USSTSTLNK.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XADRIVER.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XALINK.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XAMESSAGE.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XATEST.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XATEST.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
XIDRIVER.MAR;1	[SYSTEM]	(RWED,RWED,RWED,RE)

Total of 49 files.

# Protection for VAX/VMS System Files

Directory DB: [SYS0.SYSLIB]

ACLEDIT.INI;1	[SYSTEM]	(RWED,RWED,RWED,RE)
BASRTL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
BASRTL2.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CDDSHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CLIMAC.REQ;1	[SYSTEM]	(RWED,RWED,RWED,RE)
COBRTL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CONVSHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CRFSHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DBGSSISHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DCLTABLES.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DCXSHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DEBUG.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DELTA.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DELTA.OBJ;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DISMNTSHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DTE_DFO3.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
EDTSHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ENCRYPshr.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFCOMMON.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFLIB.TLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
ERFSHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
FDSLHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
FORDEF.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
FORIOSDEF.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
FORRTL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
IMAGELIB.OLB;1	[SYSTEM]	(RWED,RWED,RWED,RE)
IMGDMP.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LBRSHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LIB.MLB;2	[SYSTEM]	(RWED,RWED,RWED,RE)
LIB.REQ;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LIBDEF.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LIBRTL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LIBRTL2.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MOUNTSHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MTHDEF.FOR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MTHRTL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
NMLSHR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
PASRTL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
PLIRTL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
RPGRTL.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)

# Protection for VAX/VMS System Files

SCRSR. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SECURESHR. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SIGDEF. FOR;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SMBSRVSHR. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SMGSHR. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SORTSHR. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
STARLET. MLB;2	[SYSTEM]	(RWED, RWED, RWED, RE)
STARLET. OLB;2	[SYSTEM]	(RWED, RWED, RWED, RE)
STARLET. REQ;1	[SYSTEM]	(RWED, RWED, RWED, RE)
STARLETS. TLB;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SUMSHR. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
TPAMAC. REQ;1	[SYSTEM]	(RWED, RWED, RWED, RE)
TRACE. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
VMSRTL. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
XFDEF. FOR;1	[SYSTEM]	(RWED, RWED, RWED, RE)

Total of 55 files.

Directory DB: [SYSD.SYSMGR]

ACCOUNTNG. DAT;1	[SYSTEM]	(RWED, RWED, RE, )
ALFMAINT. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
LOADNET. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
LPA11STRT. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
LTLOAD. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
MAKEROOT. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
NETCONFIG. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
OPERATOR. LOG;3	[SYSTEM]	(RWED, RWED, RE, )
RTTLOAD. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
STARTNET. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SYCONFIG. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SYSHUTDWN. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SYSTARTUP. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
VMSIMAGES. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
VMSIMAGES. DAT;2	[SYSTEM]	(RWED, RWED, , )

Total of 15 files.

Directory DB: [SYSD.SYSMSG]

CLIUTLMSG. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
DBGTBKMSG. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
FILMNTMSG. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
NETWRKMSG. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
PASMSG. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
PLMSG. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
PRGDEVMSG. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
RPGMSG. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SHRIMGMSG. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SYSMTMSG. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
SYSMSG. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)

Total of 11 files.

Directory DB: [SYSD.SYSTEST]

TCNTRL. CLD;1	[SYSTEM]	(RWED, RWED, RWED, RE)
UETCLIGOO. COM;1	[SYSTEM]	(RWED, RWED, RWED, RE)
UETCLIGOO. DAT;1	[SYSTEM]	(RWED, RWED, RWED, RE)
UETCLIGOO. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)
UETCOMSOO. EXE;1	[SYSTEM]	(RWED, RWED, RWED, RE)

# Protection for VAX/VMS System Files

UETDISK00.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETDMPF00.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETDNET00.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETDNET00.DAT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETDR1W00.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETDR7800.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETFORT01.DAT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETFORT01.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETFORT02.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETFORT03.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETINIT00.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETINIT01.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLOAD00.DAT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLOAD02.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLOAD03.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLOAD04.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLOAD05.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLOAD06.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLOAD07.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLOAD08.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLOAD09.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLOAD10.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLOAD11.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETLPAK00.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETMA7800.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETMEMY01.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETNETS00.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETP.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETPHAS00.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETRFXFOR.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETSUPDEV.DAT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETTAPE00.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETTTYS00.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETUNAS00.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)

Total of 39 files.

Directory DB: [SYS0.SYSUPD]

AUTOGEN.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
BLISSREQ.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
BOOTBLDR.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
BOOTUPD.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CONSCOPY.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CVTNAF.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
CVTUAF.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DECNET.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DEVELOP.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
DXCOPY.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
EXAMPLES.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
FILETOOLS.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
HELP.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LIBDECOMP.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
LIBRARY.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)

## Protection for VAX/VMS System Files

MANAGER.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
MISCTOOLS.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
QUEUES.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
REQUIRED.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SETDEFB00.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SPKITBLD.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
STABACKIT.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
SWAPFILES.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
TEXTTOOLS.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
UETP.TLR;1	[SYSTEM]	(RWED,RWED,RWED,RE)
VMSINSTAL.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
VMSKITBLD.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
VMSKITBLD.DAT;1	[SYSTEM]	(RWED,RWED,RWED,RE)
VMSTAILOR.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
VMSUPDATE.COM;1	[SYSTEM]	(RWED,RWED,RWED,RE)
Total of 30 files.		
Directory DB:[SYSEXE]		
SYSBOOT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
Total of 1 file.		
Directory DB:[SYSEXEMIN]		
SYSBOOT.EXE;1	[SYSTEM]	(RWED,RWED,RWED,RE)
Total of 1 file.		
Grand total of 12 directories, 460 files.		



# D

## Running VAX/VMS in a C2 Environment

Many of the security features provided by VAX/VMS are directed toward the requirements for a class C2 system, as defined in the *Department of Defense Trusted Computer System Evaluation Criteria*, published by the Department of Defense Computer Security Center (CSC-STD-001-83).

This appendix describes how some of the features of VAX/VMS relate to the C2 security model and notes some specific considerations for operating a VAX/VMS system within the C2 framework. Since the terminology used in this appendix is drawn from the evaluation criteria, you should have an understanding of the above document before using this appendix.

### The Trusted Computing Base

As might be expected of this class of system, the Trusted Computing Base (TCB) provided by VAX/VMS encompasses much of the operating system. It includes the entire executive and file system, all other system components that execute in inner access modes (such as device drivers, RMS, and DCL) most system programs installed with privilege, and a variety of other utilities used by system managers to maintain data relevant to the TCB.

The objects for which VAX/VMS provides full C2-style protection are files and directories that are accessible through normal file access techniques or through global sections. VAX/VMS also provides access controls of varying levels for other objects such as devices, logical name tables, and queues, but lacks the full set of features required by the C2 criteria (principally auditing of accesses).

### Protecting the TCB

The code and data that make up the VMS TCB reside in files and, in part, in the address space of the running operating system. Their integrity is protected by the use of file access controls and memory page protection. Memory page protection is set up by VMS as it executes and is normally not of concern to the system manager. The files containing the TCB are by default correctly protected when VMS is installed; however, the protection can be altered by sufficiently privileged users. Appendix C of this handbook describes the correct file protection of all the VMS components.

Certain privileges allow their holder to bypass normal file and memory access controls either directly or indirectly and, therefore, must not be granted to persons other than the system manager, security officer, or other very trusted persons. These privileges are described in detail in Appendix A of this handbook. Table 5-5 in Chapter 5 of this handbook summarizes the privileges and categorizes them in terms of their sensitivity.

Privileges in the FILES and ALL categories allow the holder to violate the integrity of the TCB. Privileges in the SYSTEM category allow the holder to interfere with normal system operation and cause denial of service; however, they do not allow the holder to actually violate object access controls. Some privileges in the SYSTEM category also permit certain forms of spoofing that could ultimately result in violations of access controls. Privileges in the DEVOUR and GROUP categories permit their holder to consume resources without limit, thereby causing possible denial of service, and interference with the operations of others in the same group. The GRPPRV privilege, in particular, permits the holder to violate normal access controls within his group.

### Individual Accountability

The proper use of usernames, UICs, and passwords ensures that individual accountability is enforced by VAX/VMS. The following practices and features, however, result in the loss of individual accountability and must not be used in a C2 environment:

- 1 Assigning the same UIC to more than one user. The UIC is used as the universal internal user identifier; therefore, unique UICs must be assigned to all users.



- 2 Open accounts. Lack of a password makes an account available to all users aware of its identity. The system manager can prevent open accounts by (a) not setting null passwords with AUTHORIZE, and (b) ensuring that all accounts are set up with a non-zero minimum password length.
- 3 Group accounts. Individual accountability is lost when multiple persons share an account. Each user must be given his or her own unique account.
- 4 Autologin. The autologin feature must not be used because it associates an account with a particular terminal instead of a particular person and, therefore, causes individual accountability to be lost.
- 5 Network proxy accounts for groups. In order to preserve individual accountability, each individual in a network must be given his own unique network proxy account on each node to which he has any access. This practice is normally best managed by assigning the same username and UIC on all applicable nodes to a person and setting up individual proxies among the corresponding accounts.

### Object Protection and Reuse

File and directory protection are discussed extensively in Chapter 4 of this handbook. A series of mechanisms, described in Section 4.5, provide useful default protection for newly created objects.

Reuse of system memory pages is protected by the memory management subsystem and cannot be defeated. Reuse of disk blocks is protected by the highwater marking and erase on delete features, which are described in Section 5.6. To conform to C2 criteria, all system disk volumes must have highwater marking enabled, which is the default.

VAX/VMS considers magnetic tapes to be single user devices. Tape protection is only available at the volume level. In other words, an entire volume may be assigned ownership and protection but not the individual files it contains. Because of this policy, VAX/VMS provides no protection against reuse of tape. Tapes that are recycled to new users must be externally erased by operations personnel.

### Protection of the Audit Trail

The security audit trail is recorded in the operator log file and on terminals enabled as security operators. The operator log is normally protected against reading or modification by unauthorized users.

So that traces of system penetrations cannot be fully erased, you can further protect the audit log by adopting the following measures:

- 1 Place, in a physically secure location, a hardcopy terminal enabled as a security operator.
- 2 Protect the audit log file with the following audit measures:

- a Enable, at minimum, audits on ACL and audit events with the following command:

```
$ SET AUDIT /ALARM /ENABLE=(ACL,AUDIT)
```

- b Place on the operator log file an ACL entry that enables auditing of all accesses for modification and deletion.

```
$ SET FILE SYS$MANAGER:OPERATOR.LOG -  
/ACL=(ALARM=SECURITY,ACCESS=WRITE+DELETE+CONTROL+SUCCESS)
```

These audits ensure that any attempt to tamper with the audit log will result either in the system audit controls left obviously turned off, or with one last "footprint" in the audit log. While these measures are not completely airtight, circumventing them requires extensive programming to completely bypass the file system or to rummage about the operating system's innards.

### Auditing Actions of a System Operator or Administrator

All actions taken by trusted users of the system (operators, administrators, security officers, etc.) can be audited by the enforced use of terminal session auditing as described in Section 5.8.5. Attempts to defeat the auditing can be detected by measures similar to those used to protect the audit log:

- 1 Enable auditing of authorization modifications.

- 2 Place ACL entries on the captive login command procedures and the directories containing them to detect modification of the procedures.

### Documentation

This handbook, together with the applicable reference documentation, constitutes the *Trusted Facility Manual* for use by the system administrator. The first four chapters of this handbook constitute the *Security Features User's Guide* and should be made available to all system users.

### Physical Security

As has been pointed out elsewhere in this guide, physical and environmental security are critical to the secure operation of the system. Preventing theft of media and output is an obvious consideration. In addition, the console terminal must always be physically secured because it controls operation of the CPU and, consequently, operation of the system.

### Configuration Guidelines

The security features described in this guide apply to most VAX configurations. They are supported by all VAX CPUs, including MicroVAX CPUs, and apply to all supported mass storage and communications devices.

VAXcluster configurations also fully support the security features. A VAXcluster is considered a single security and management domain and normally operates with a shared authorization database. For more information, please refer to the *Guide to VAXclusters*.

VAX/VMS does not meet the needs of the C2 requirements in configurations in which an MA780 shared memory subsystem is used as a shared memory among two or more independent processors. The communications objects in the shared memory (common event flag clusters, mailboxes, and global sections) do not support access control lists or security auditing. However, when a VAX-11/782 dual processor system uses the MA780 as the main memory, all security features are fully supported.

## Running VAX/VMS in a C2 Environment

The evaluation criteria do not address network operation. However, when connected to a DECnet network, VAX/VMS provides security commensurate with the security of the base operating system if the following restrictions are met.

- 1** All operating systems connected to the network are VAX/VMS systems or systems of equivalent security and are systems administered in a secure manner.
- 2** Default accounts are not provided for file and general task access. Limiting access to explicit access control strings and proxy access preserves individual accountability.
- 3** The communications lines are secured from wiretaps by use of link encryption devices or by physical security.

# E

## Alarm Messages

This appendix describes the alarm messages that result from auditing various system events.

### E.1 Alarms Auditing Access to Files and Global Sections

You can audit successful or unsuccessful access to a file or global section by specifying the FILE\_ACCESS keyword with the /ENABLE qualifier of the SET AUDIT command. READ, WRITE, EXECUTE, DELETE, or CONTROL access modes can be audited. You can also audit successful access to a file or global section through the use of GRPPRV, READALL, SYSPRV, or BYPASS privilege.

In addition to the information contained in all alarm messages, this type of alarm contains the following:

- The name of the file or global section accessed
- The mode of access
- The image used to access the file or global section
- The privileges used to access the file or global section

The following alarm messages are examples of the file and global section access alarms.

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:19:49.95 XXXXXXXXXXXX
Security alarm on LASSIE / Successful file access
Time: 15-JUN-1985 12:19:49.94
PID: 26C0062B
User Name: WILSON
Image: LASSIE$DMAO:[SYSTEM.SYSEX]:TYPE.EXE
File: _LASSIE$DMAO:[TIMMY]PRIVILEGE.CMD;1
Mode: READ
Privs Used: (None)
```

## Alarm Messages

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:20:14.61 XXXXXXXXXXXX
Security alarm on LASSIE / File access failure
Time: 15-JUN-1985 12:20:14.60
PID: 26C0062B
User Name: WILSON
Image: LASSIE$DMAO: [SYSTEM.SYSEX] :TYPE.EXE
File: _LASSIE$DMAO: [TIMMY]PRIVILEGE.CMD;1
Mode: READ
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:22:01.17 XXXXXXXXXXXX
Security alarm on LASSIE / Successful global section access
Time: 15-JUN-1985 12:22:01.16
PID: 00000112
User Name: SYSTEM
Image: LASSIE$DMAO: [SYSO.] [SYSEX]MAIL.EXE
Global Section: SMG$TERMTABLE
File: _LASSIE$DMAO: [SYSO.] [SYSEX]TERMTABLE.EXE;1
Mode: READ
Privs Used: (None)
```

```
XXXXXXXXXX OPCOM 18-APR-1985 10:11:40.21 XXXXXXXXXXXX
Security alarm on VENUS / Global section access failure
Time: 18-APR-1985 10:11:40.20
PID: 000000A5
User Name: SYSTEM
Image: VENUS$DMAO: [SYSO.] [SYSMGR]PRIV_USERS.EXE;3
Global Section: ACLEDT_002
File: _VENUS$DMAO: [SYSO.] [SYSEX]ACLEDT.EXE;1
Mode: READ
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:19:49.95 XXXXXXXXXXXX
Security alarm on LASSIE / Successful file access
Time: 15-JUN-1985 12:19:49.94
PID: 26C0062B
User Name: WILSON
Image: LASSIE$DMAO: [SYSO.] [SYSEX]TYPE.EXE
File: _LASSIE$DMAO: [TIMMY.CMD]PRIVILEGE.CMD;1
Mode: READ
Privs Used: BYPASS
```

---

## E.2 Alarms Requested by an ACL

You can audit successful or unsuccessful access to a file. To do so, you add an ALARM\_JOURNAL ACE to a file's ACL and then enable ACL alarms by specifying the ACL keyword with the /ENABLE qualifier of the SET AUDIT command. For example, the following ACE in a file's ACL requests that an alarm occur whenever the file is successfully read.

```
(ALARM_JOURNAL=SECURITY,ACCESS=SUCCESS+READ)
```

However, the ACL alarm has no effect unless it is enabled by the following command:

## Alarm Messages

\$ SET AUDIT/ALARM/ENABLE=(ACL)

READ, WRITE, EXECUTE, DELETE, or CONTROL modes can be audited. In addition to the information contained in all alarm messages, this type of alarm contains the following:

- The name of the file or global section accessed
- The mode of access
- The image used to access the file or global section
- The privileges used to access the file or global section

The following alarm messages are examples of alarms requested by an ACL.

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:19:49.95 XXXXXXXXXXXX
Security alarm on LASSIE / Successful file access
Time: 15-JUN-1985 12:19:49.94
PID: 26C0062B
User Name: MENACE
Image: LASSIE$DMAO:[SYSO.][SYSEXE]TYPE.EXE
File: _LASSIE$DMAO:[TIMMY]PRIVATE.LIS;1
Mode: READ
Privs Used: (None)
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:20:14.61 XXXXXXXXXXXX
Security alarm on LASSIE / File access failure
Time: 15-JUN-1985 12:20:14.60
PID: 26C0062B
User Name: MENACE
Image: LASSIE$DMAO:[SYSO.][SYSEXE]TYPE.EXE
File: _LASSIE$DMAO:[TIMMY]PRIVATE.LIS;1
Mode: READ
```

---

### E.3 Alarms Auditing INSTALL Operations

You can audit the use of the Install Utility (to install an image or to remove an installed image) by specifying the INSTALL keyword with the /ENABLE qualifier of the SET AUDIT command. In addition to the information contained in all alarm messages, this type of alarm contains:

- The type of INSTALL operation
- The name of the image affected by the INSTALL operation
- The flags set by INSTALL operation
- The privileges used in the INSTALL operation

## Alarm Messages

The following alarm messages are examples of alarms resulting from INSTALL operations.

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:37:49.69 XXXXXXXXXXXX
Security alarm on LASSIE / Installed file addition
Time: 15-JUN-1985 12:37:49.68
PID: 00000113
User Name: SYSTEM
File: LASSIE$DMAO:[SYSO.][SYSEXE]NCP.EXE;9
INSTALL Flags: OPEN HEADER_RESIDENT
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:37:54.20 XXXXXXXXXXXX
Security alarm on LASSIE / Installed file removal
Time: 15-JUN-1985 12:37:54.18
PID: 00000113
User Name: SYSTEM
File: LASSIE$DMAO:[SYSO.][SYSEXE]NCP.EXE;9
INSTALL Flags: OPEN HEADER_RESIDENT
```

```
XXXXXXXXXX OPCOM 3-MAY-1985 10:12:19.18 XXXXXXXXXXXX
Security alarm on VENUS / Installed file addition
Time: 03-MAY-1985 10:12:19.11
PID: 24E00A03
User Name: SMITH
File: DUA8:[SYS8.SYSCOMMON.SYSEXE]SDA.EXE;1
INSTALL Flags: PRIVILEGED
Privileges: CMKRNL
```

```
XXXXXXXXXX OPCOM 3-MAY-1985 10:12:27.84 XXXXXXXXXXXX
Security alarm on VENUS / Installed file removal
Time: 03-MAY-1985 10:12:27.83
PID: 24E00A03
User Name: SMITH
File: DUA8:[SYS8.SYSCOMMON.SYSEXE]SDA.EXE;1
INSTALL Flags: PRIVILEGED
Privileges: CMKRNL
```

---

### E.4 Alarms Resulting from Modifications to the Rights Database

You can audit any changes made to the rights database by specifying the AUTHORIZATION keyword with the /ENABLE qualifier of the SET AUDIT command. The types of changes that you can audit are:

- The creation of a new rights database
- The addition of an identifier
- The removal of an identifier
- The modification of an identifier
- The granting of an identifier to a holder



## Alarm Messages

- The revoking an identifier from a holder
- The modification of a holder

In addition to the information contained in all alarm messages, this type of alarm contains the following:

- The image used to modify the rights database
- The change made to the rights database

The following alarm messages are examples of alarms resulting from modification of the rights database.

```
XXXXXXXXXX OPCOM 25-APR-1985 10:42:34.42 XXXXXXXXXXXX
Security alarm on VENUS / Rights database created
Time: 25-APR-1985 10:42:34.39
PID: 22200150
User Name: SMITH
Image: DUA8:[SYS8.][SYSCOMMON.SYSEXE]AUTHORIZE.EXE
File: SYS$SYSTEM:RIGHTSLIST.DAT;1
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:26:44.60 XXXXXXXXXXXX
Security alarm on LASSIE / Identifier added
Time: 15-JUN-1985 12:26:44.01
PID: 00000113
User Name: SYSTEM
Image: LASSIE$DMAO:[SYSO.][SYSEXE]AUTHORIZE.EXE
ID name: TIMMY
ID value: %X80010000
Id attributes: (None)
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:33:23.06 XXXXXXXXXXXX
Security alarm on LASSIE / Identifier removed
Time: 15-JUN-1985 12:33:23.05
PID: 00000113
User Name: SYSTEM
Image: LASSIE$DMAO:[SYSO.][SYSEXE]AUTHORIZE.EXE
ID name: MASTERS
ID value: %X80010000
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:27:17.44 XXXXXXXXXXXX
Security alarm on LASSIE / Identifier modified
Time: 15-JUN-1985 12:27:17.36
PID: 00000113
User Name: SYSTEM
Image: LASSIE$DMAO:[SYSO.][SYSEXE]AUTHORIZE.EXE
ID name: ROBINSON
ID value: %X80010000
Id attributes: RESOURCE
```

## Alarm Messages

XXXXXXXXXX OPCOM 6-MAY-1985 10:14:59.42 XXXXXXXXXXXX  
Security alarm on VENUS / Identifier modified  
Time: 06-MAY-1985 10:14:59.32  
PID: 20A00385  
User Name: SMITH  
Image: VENUS\$DUB44: [SYSO.] [SYSCOMMON.SYSEXE] AUTHORIZE.EXE  
ID name: DEMILO  
ID value: %X800186A0  
New ID value: %X800186A1

XXXXXXXXXX OPCOM 6-MAY-1985 10:15:12.02 XXXXXXXXXXXX  
Security alarm on VENUS / Identifier modified  
Time: 06-MAY-1985 10:15:12.02  
PID: 20A00385  
User Name: SMITH  
Image: VENUS\$DUB44: [SYSO.] [SYSCOMMON.SYSEXE] AUTHORIZE.EXE  
ID name: BONNELL  
ID value: %X800186A1  
New ID name: SMITH

XXXXXXXXXX OPCOM 15-JUN-1985 12:28:19.54 XXXXXXXXXXXX  
Security alarm on LASSIE / Identifier granted  
Time: 15-JUN-1985 12:28:19.52  
PID: 00000113  
User Name: SYSTEM  
Image: LASSIE\$DMAO: [SYSO.] [SYSEXE] AUTHORIZE.EXE  
ID name: PERSONNEL  
ID value: %X80010000  
Holder name: PERKINS  
Holder UIC: [214,202]  
Id attributes: (None)

XXXXXXXXXX OPCOM 15-JUN-1985 12:28:49.61 XXXXXXXXXXXX  
Security alarm on LASSIE / Identifier revoked  
Time: 15-JUN-1985 12:28:49.60  
PID: 00000113  
User Name: SYSTEM  
Image: LASSIE\$DMAO: [SYSO.] [SYSEXE] AUTHORIZE.EXE  
ID name: PERSONNEL  
ID value: %X80010000  
Holder name: MANSON  
Holder UIC: [214,210]

XXXXXXXXXX OPCOM 15-JUN-1985 12:28:19.54 XXXXXXXXXXXX  
Security alarm on LASSIE / ID holder modified  
Time: 15-JUN-1985 12:28:19.52  
PID: 00000113  
User Name: SYSTEM  
Image: LASSIE\$DMAO: [SYSO.] [SYSEXE] AUTHORIZE.EXE  
ID name: PAYROLL  
ID value: %X80010000  
Holder name: ARNOLD  
Holder UIC: [220,133]  
Id attributes: RESOURCE

## **E.5 Alarms Resulting from Changes to System or Network UAF**

Specifying the AUTHORIZATION keyword with the /ENABLE qualifier of the SET AUDIT command enables auditing of changes made to the system or network UAF in addition to auditing changes to the rights database. The types of changes that you can audit are:

- Adding a system UAF record
- Deleting a system UAF record
- Changing a system UAF record
- Copying a system UAF record
- Renaming a system UAF record
- Adding a network UAF record
- Deleting a network UAF record
- Modifying a network UAF record

In addition to the information contained in all alarm messages, this type of alarm contains the following:

- The record that was modified
- The change that was made

The following alarm messages are examples of alarms resulting from modification of the system or network UAF.

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:27:37.03 XXXXXXXXXXXX
Security alarm on LASSIE / System UAF record addition
Time: 15-JUN-1985 12:27:36.97
PID: 00000113
User Name: SYSTEM
Rec Add: COOPER
Fields Mod: FLAGS PWDLIFETIME
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:33:28.36 XXXXXXXXXXXX
Security alarm on LASSIE / System UAF record deletion
Time: 15-JUN-1985 12:33:28.35
PID: 00000113
User Name: SYSTEM
Rec Del: MELVILLE
```

## Alarm Messages

XXXXXXXXXX OPCOM 15-JUN-1985 12:27:52.26 XXXXXXXXXXXX  
Security alarm on LASSIE / System UAF record modification  
Time: 15-JUN-1985 12:27:52.25  
PID: 23C00155  
User Name: MENACE  
Rec Mod: GOWER  
Fields Mod: PRIVILEGES

XXXXXXXXXX OPCOM 15-JUN-1985 12:32:28.50 XXXXXXXXXXXX  
Security alarm on LASSIE / System UAF record copied  
Time: 15-JUN-1985 12:32:28.49  
PID: 00000113  
User Name: SYSTEM  
Rec Add: MUTT  
From Name: JEFF  
Fields Mod: (None)

XXXXXXXXXX OPCOM 15-JUN-1985 12:32:40.48 XXXXXXXXXXXX  
Security alarm on LASSIE / System UAF record renamed  
Time: 15-JUN-1985 12:32:40.47  
PID: 00000113  
User Name: SYSTEM  
New Name: ALIAS  
Old Name: SILAS  
Fields Mod: (None)

XXXXXXXXXX OPCOM 15-JUN-1985 12:34:13.84 XXXXXXXXXXXX  
Security alarm on LASSIE / Network UAF record addition  
Time: 15-JUN-1985 12:34:13.77  
PID: 00000113  
User Name: SYSTEM  
Rec Add: VENUS::SYSTEM          SYSTEM

XXXXXXXXXX OPCOM 15-JUN-1985 12:36:23.96 XXXXXXXXXXXX  
Security alarm on LASSIE / Network UAF record deletion  
Time: 15-JUN-1985 12:36:23.95  
PID: 00000113  
User Name: SYSTEM  
Rec Del: GEORGE::SYSTEM          SYSTEM

XXXXXXXXXX OPCOM 18-APR-1985 09:50:05.12 XXXXXXXXXXXX  
Security alarm on VENUS / Network UAF record modification  
Time: 18-APR-1985 09:50:04.11  
PID: 000000A5  
User Name: SYSTEM  
New Rec: MARS::USER          SUESS  
Old Rec: MARS::USER          WIMBLY

---

### E.6 Alarms Resulting from Password Changes

The AUTHORIZATION keyword specified with the /ENABLE qualifier of the SET AUDIT command also enables auditing of changes to a user or the system password. In addition to the information contained in all alarm messages, this type of alarm specifies which password was changed.

The following alarm messages are examples of alarms resulting from modification of the system or network UAF.

```
XXXXXXXXXXXX OPCOM 15-JUN-1985 12:27:52.26 XXXXXXXXXXXX
Security alarm on LASSIE / System UAF record modification
Time: 15-JUN-1985 12:27:52.25
PID: 20C00133
User Name: MENACE
Rec Mod: DENNIS
Fields Mod: PASSWORD
```

```
XXXXXXXXXXXX OPCOM 3-MAY-1985 14:17:33.98 XXXXXXXXXXXX
Security alarm on VENUS / System UAF record modification
Time: 03-MAY-1985 14:17:33.90
PID: 24E00A03
User Name: MENACE
Rec Mod: ** SYSPWD **
Fields Mod: PASSWORD
```

---

### E.7 Breakin Attempt Alarms

You can audit breakin attempts by specifying the BREAKIN keyword with the /ENABLE qualifier of the SET AUDIT command. You can audit the DIALUP, LOCAL, REMOTE, NETWORK and DETACHED breakin types.

In addition to the information contained in all alarm messages, this type of alarm contains the following:

- The type of breakin attempt
- The password used
- The device used
- The origin of attempt if the breakin type was REMOTE or NETWORK
- The parent username if the breakin type was DETACHED

## Alarm Messages

The following alarm messages are examples of the breakin attempt alarms.

```
XXXXXXXXXX OPCOM 15-JUN-1985 14:33:20.69 XXXXXXXXXXXX
Security alarm on LASSIE / Dialup interactive breakin detection
Time: 15-JUN-1985 14:33:20.67
PID: 00000052
User Name: SNIDELY
Password: EASYDOESIT
Dev Name: _TTA1:
```

```
XXXXXXXXXX OPCOM 15-APR-1985 14:32:58.79 XXXXXXXXXXXX
Security alarm on LASSIE / Local interactive breakin detection
Time: 15-APR-1985 14:32:58.77
PID: 00000051
User Name: SNIDELY
Password: MUMBLE
Dev Name: _TTA1:
```

```
XXXXXXXXXX OPCOM 17-APR-1985 14:54:34.87 XXXXXXXXXXXX
Security alarm on LASSIE / Remote interactive breakin detection
Time: 17-APR-1985 14:54:34.83
PID: 0000005D
User Name: FAGAN
Password: LIGHTLY
Dev Name: _RTA1:
Source: 2.218 VENUS::SYSTEM
```

```
XXXXXXXXXX OPCOM 17-APR-1985 14:55:28.51 XXXXXXXXXXXX
Security alarm on LASSIE / Network breakin detection
Time: 17-APR-1985 14:55:28.50
PID: 0000005E
User Name: DECNET
Password: FARFARAWAY
Source: 2.218 VENUS::SYSTEM
```

```
XXXXXXXXXX OPCOM 18-APR-1985 16:14:59.72 XXXXXXXXXXXX
Security alarm on LASSIE / Detached process breakin detection
Time: 18-APR-1985 16:14:59.60
PID: 00000162
User Name: ARTFUL
Password: DODGER
Parent U.N.: SYSTEM
```

---

## E.8 Login Alarms

You can audit successful logins by specifying the LOGIN keyword with the /ENABLE qualifier of the SET AUDIT command. You can audit BATCH, DIALUP, LOCAL, REMOTE, NETWORK, SUBPROCESS and DETACHED login types.

In addition to the information contained in all alarm messages, this type of alarm contains the following:

- The type of login
- The device used
- The origin of the login if it was REMOTE or NETWORK
- The parent PID if the login was SUBPROCESS
- The parent username if the login was DETACHED

The following alarm messages are examples of the successful login alarms.

```
XXXXXXXXXX OPCOM 17-APR-1985 14:57:19.25 XXXXXXXXXXXX
Security alarm on TIGER / Batch process login
Time: 17-APR-1985 14:57:19.24
PID: 320B005F
User Name: MENACE
```

```
XXXXXXXXXX OPCOM 15-APR-1985 14:34:33.68 XXXXXXXXXXXX
Security alarm on TIGER / Dialup interactive login
Time: 15-APR-1985 14:34:33.67
PID: 00030055
User Name: GRAHAM
Dev Name: _TTA1:
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:53:01.56 XXXXXXXXXXXX
Security alarm on LASSIE / Local interactive login
Time: 15-JUN-1985 12:53:01.55
PID: 03500124
User Name: JUNE
Dev Name: _LTA7:
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:52:38.85 XXXXXXXXXXXX
Security alarm on LASSIE / Remote interactive login
Time: 15-JUN-1985 12:52:38.84
PID: 00000123
User Name: SYSTEM
Dev Name: _RTA1:
Source: 2.91 LASSIE::SYSTEM
```

## Alarm Messages

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:50:25.93 XXXXXXXXXXXX
Security alarm on LASSIE / Network login
Time: 15-JUN-1985 12:50:25.92
PID: 00000121
User Name: SYSTEM
Source: 2.58 TIGER::2E0004A4
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:49:54.92 XXXXXXXXXXXX
Security alarm on LASSIE / Subprocess login
Time: 15-JUN-1985 12:49:54.91
PID: 00000120
User Name: ADAM
Parent PID: 00000113
```

```
XXXXXXXXXX OPCOM 17-APR-1985 17:08:08.34 XXXXXXXXXXXX
Security alarm on TIGER / Detached process login
Time: 17-APR-1985 17:08:08.31
PID: 00000093
User Name: ISSAC
Parent U.N.: ABRAHAM
```

---

### E.9 Login Failure Alarms

You can audit login failures by specifying the LOGFAILURE keyword with the /ENABLE qualifier of the SET AUDIT command. You can audit the BATCH, DIALUP, LOCAL, REMOTE, NETWORK, SUBPROCESS and DETACHED login failure types.

In addition to the information contained in all alarm messages, this type of alarm contains the following:

- The type of login
- The device used
- The status detailing the reason for the failure
- The origin of the login if it was REMOTE or NETWORK
- The parent PID if the login was SUBPROCESS
- The parent username if the login was DETACHED

The following alarm messages are examples of the login failure alarms.

```
XXXXXXXXXX OPCOM 17-APR-1985 15:03:30.86 XXXXXXXXXXXX
Security alarm on TIGER / Batch process login failure
Time: 17-APR-1985 15:03:30.78
PID: 00030060
User Name: RUBENS
Status: %LOGIN-F-BADDAY, you are not authorized to login today
```



```
XXXXXXXXXXXX OPCOM 15-APR-1985 14:35:30.77 XXXXXXXXXXXX
Security alarm on TIGER / Dialup interactive login failure
Time: 15-APR-1985 14:35:30.47
PID: 00000057
User Name: LILY
Status: %LOGIN-F-NOSUCHUSER, no such user
Dev Name: _TTA1:
```

```
XXXXXXXXXXXX OPCOM 15-JUN-1985 12:46:36.82 XXXXXXXXXXXX
Security alarm on LASSIE / Local interactive login failure
Time: 15-JUN-1985 12:46:36.80
PID: 0000011C
User Name: TIMMY
Status: %LOGIN-F-NOSUCHUSER, no such user
Dev Name: _LTA6:
```

```
XXXXXXXXXXXX OPCOM 15-JUN-1985 12:40:50.40 XXXXXXXXXXXX
Security alarm on LASSIE / Remote interactive login failure
Time: 15-JUN-1985 12:40:50.24
PID: 00000115
User Name: THOMPSON
Status: %LOGIN-F-NOSUCHUSER, no such user
Dev Name: _RTA1:
Source: 2.91 LASSIE::SYSTEM
```

```

XXXXXXXXXXXX OPCOM 15-JUN-1985 12:48:43.50 XXXXXXXXXXXX
Security alarm on LASSIE / Network login failure
Time: 15-JUN-1985 12:48:43.49
PID: 0000011D
User Name: DECNET
Status: %LOGIN-F-NOSUCHUSER, no such user
Source: 2.58 MILOS::SMITH

```

```
XXXXXXXXXX OPCOM 18-APR-1985 17:07:54.60 XXXXXXXXXXXX
Security alarm on LASSIE / Subprocess login failure
Time: 18-APR-1985 17:07:54.59
PID: 00000195
User Name: RANGER
Status: %LOGIN-F-OUTPUTERR, error opening primary
        output file SYS$OUTPUT
Parent PID: 0000018D
```

```

XXXXXXXXXXXX OPCOM 15-JUN-1985 12:49:30.69 XXXXXXXXXXXX
Security alarm on LASSIE / Detached process login failure
Time: 15-JUN-1985 12:49:30.68
PID: 0000011E
User Name: PUPPY
Status: %SYSTEM-F-NOLOGNAM, no logical name match
Parent U.N.: DOGGIE

```

---

## E.10 Logout Alarms

You can audit logouts by specifying the LOGOUT keyword with the /ENABLE qualifier of the SET AUDIT command. You can audit BATCH, DIALUP, LOCAL, REMOTE, NETWORK, SUBPROCESS and DETACHED logout types.

In addition to the information contained in all alarm messages, this type of alarm contains the following:

- The type of logout
- The device used
- The origin of the login if it was REMOTE or NETWORK
- The parent PID if the login was SUBPROCESS

The following alarm messages are examples of the logout alarms.

```
XXXXXXXXXX OPCOM 17-APR-1985 14:57:21.11 XXXXXXXXXXXX
Security alarm on TIGER / Batch process logout
Time: 17-APR-1985 14:57:21.10
PID: 0000005F
User Name: LILY
```

```
XXXXXXXXXX OPCOM 15-APR-1985 14:35:15.78 XXXXXXXXXXXX
Security alarm on TIGER / Dialup interactive logout
Time: 15-APR-1985 14:35:15.77
PID: 00000056
User Name: LILY
Dev Name: _TTA1:
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:53:07.09 XXXXXXXXXXXX
Security alarm on LASSIE / Local interactive logout
Time: 15-JUN-1985 12:53:07.08
PID: 00000124
User Name: TIMMY
Dev Name: _LTA7:
```

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:52:41.23 XXXXXXXXXXXX
Security alarm on LASSIE / Remote interactive logout
Time: 15-JUN-1985 12:52:41.22
PID: 00000123
User Name: TIMMY
Dev Name: _RTA1:
Source: 2.91 LASSIE::TIMMY
```

## Alarm Messages

```
XXXXXXXXXX OPCOM 17-APR-1985 14:44:22.23 XXXXXXXXXXXX
Security alarm on VENUS / Network logout
Time: 17-APR-1985 14:44:22.22
PID: 00000058
User Name: DEMILO
Source: 2.91 LASSIE::SYSTEM

XXXXXXXXXX OPCOM 15-JUN-1985 12:49:59.49 XXXXXXXXXXXX
Security alarm on LASSIE / Subprocess logout
Time: 15-JUN-1985 12:49:59.48
PID: 00000120
User Name: TIMMY
Parent PID: 00000113

XXXXXXXXXX OPCOM 17-APR-1985 17:08:09.09 XXXXXXXXXXXX
Security alarm on VENUS / Detached process logout
Time: 17-APR-1985 17:08:09.08
PID: 00000093
User Name: DEMILO
```

---

### E.11 Volume Mount and Dismount Alarms

You can audit login failures by specifying the MOUNT keyword with the /ENABLE qualifier of the SET AUDIT command. In addition to the information contained in all alarm messages, this type of alarm contains the following:

- The image used to mount the volume
- The device used
- The log file recording the operation
- The volume name, UIC, and protection
- The flags set during the operation

The following alarm message is an example of the mount alarm.

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:57:00.21 XXXXXXXXXXXX
Security alarm on LASSIE / Volume mount
Time: 15-JUN-1985 12:56:59.93
PID: 00000113
User Name: SYSTEM
Image: LASSIE$DMAO:[SYSO.][SYSEX]VMOUNT.EXE;1
Dev Name: _LASSIE$CSA1:
Log Name: DISK$CONSOLE
Vol Name: CONSOLE
Vol UIC: [1,4]
Vol Pro: S:RWED,O:RWED,G,W
Mount Flags: FOREIGN MESSAGE SYSTEM
```

## Alarm Messages

The MOUNT keyword specified with the /ENABLE qualifier of the SET AUDIT command also enables auditing of dismount operations. In addition to the information contained in all alarm messages, this type of alarm contains the following:

- The image used to dismount the volume
- The device used
- The log file recording the operation
- The volume name
- The flags set during the operation

The following alarm message is an example of the dismount alarm.

```
XXXXXXXXXX OPCOM 15-JUN-1985 12:57:06.78 XXXXXXXXXXXX
Security alarm on LASSIE / Volume dismount
Time: 15-JUN-1985 12:57:06.77
PID: 00000113
User Name: SYSTEM
Image: LASSIE$DMAO:[SYSO.][SYSEXE]DISMOUNT.EXE;1
Dev Name: _LASSIE$CSA1:
Log Name: DISK$CONSOLE
Vol Name: CONSOLE
Mount Flags: (None)
```

---

### E.12 Alarms Resulting from Execution of SET AUDIT Command

You can enable auditing of any execution of the SET AUDIT command by specifying the AUDIT keyword with the /ENABLE qualifier of the SET AUDIT command. By using this type of alarm, you audit the use of alarms.

The following alarm messages are examples of audit alarms.

```
XXXXXXXXXX OPCOM 3-MAY-1985 10:11:19.42 XXXXXXXXXXXX
Security alarm on VENUS / Security audit alarms enabled
Time: 03-MAY-1985 10:11:19.41
PID: 24E00A03
User Name: SMITH
Audits Modified:
INSTALL
```

## Alarm Messages

XXXXXXXXXX OPCOM 3-MAY-1985 10:12:58.84 XXXXXXXXXXXX  
Security alarm on VENUS / Security audit alarms disabled  
Time: 03-MAY-1985 10:12:58.83  
PID: 24E00A03  
User Name: SMITH  
Audits Modified:  
INSTALL  
AUDIT



---

## Glossary

**Access Control List:** A list that defines the kinds of access to be granted or denied to users of an object. Access control lists can be created for objects such as files, devices, and mailboxes. Each access control list consists of one or more entries known as access control list entries.

**Access Control List Entry:** An entry in an access control list. Access control list entries may specify identifiers and the access rights to be granted or denied the holders of the identifiers, default protection for directories, or security alarm details. Access control lists for each object can hold numerous entries, limited only by overall space and performance considerations.

**ACE:** *Access Control List Entry*

**ACL Editor:** A VAX/VMS utility that helps users to create and maintain access control lists.

**ACL:** *Access Control List*

**Alarm:** *Security Alarm*

**Alphanumeric UIC:** A format of user identification code (UIC) that specifies the user's group and member in alphanumeric form rather than numeric form.

**Attribute:** In the security context, an attribute is a field of information maintained in the rights database that identifies some characteristic accorded to all holders of the identifier. For example, if an identifier possesses the resource attribute, holders of that identifier are able to charge resources such as disk space usage to that identifier.

**Auditing:** The act of noting the occurrence of an event that has security implications.

## Glossary

**Authentication:** The act of establishing the identity of users when they start to use the system. VAX/VMS (and most other commercial operating systems) use passwords as the primary authentication mechanism.

**Breach:** A break in the system security that results in admittance of a person or program to an object.

**Breakin Attempt:** An effort made by an unauthorized source to gain access to the system. Since the first system access is achieved through logging in, breakin attempts primarily refer to attempts to login illegally. These attempts focus on supplying passwords for users known to have accounts on the system, through informed guesses or other trial-and-error methods.

**Captive Accounts:** A type of VAX/VMS account that limits the activities of the user. Typically, the user is restricted to using certain command procedures and/or commands. The user may not be allowed to use the CTRL/Y key. (This type of account is synonymous with a turnkey or tied account.)

**Decryption:** The process that restores encoded information to its original unencoded form.

**Discretionary Controls:** Security controls that are applied at the user's option, that is, they are not required. Access control lists are typical of such optional security features. Discretionary controls are the opposite of mandatory controls.

**Disk Scavenging:** A term that refers to any method of obtaining information from a disk that the owner intended to discard. The information, although no longer accessible to the original owner by normal means, retains a sufficient amount of its original magnetic encoding that it can be retrieved and used by one of the scavenging methods.

**Encryption:** A process of encoding information so that its content is no longer immediately obvious to anyone who obtains a copy of it.

**Erasur Pattern:** A character string that can be used to overwrite magnetic media for the purpose of erasing the information that was previously stored in that area.



**Erase-on-allocate:** A technique that applies an erasure pattern whenever a new area is allocated for a file's extent. The new area is erased with the erasure pattern so that subsequent attempts to read the area can only yield the erasure pattern and not some valuable remaining data. This technique is used to discourage disk scavenging.

**Erase-on-delete:** A technique that applies an erasure pattern whenever a file is deleted or purged. This technique is used to discourage disk scavenging.

**Evasive Action:** A responsive behavior by VAX/VMS to discourage breakin attempts whenever they seem to be in progress. VAX/VMS has a set of criteria it uses to detect the fact that breakin attempts may be underway. Typically, once VAX/VMS becomes suspicious that an unauthorized user is attempting to login, the evasive action consists of locking out all login attempts by the offender for a limited period of time.

**General Identifier:** One of three possible types of identifiers that specify one or more groups of users. The general identifier is alphanumeric and typically is a convenient term that symbolizes the nature of the group of users. For example, typical general identifiers might be PAYROLL for all users allowed to run payroll applications, RESERVATIONS for operators at the reservations desk, or CHEM224 for students in the CHEMISTRY 224 class, and so forth.

**Highwater Marking:** A technique for discouraging disk scavenging. In the truest sense, this technique tracks the furthest extent that the owner of a file has written into the file's allocated area. It then prohibits any attempts at reading beyond the written area, on the premise that any information that exists beyond the currently written limit is information some user had intended to discard. VAX/VMS accomplishes the goals of highwater marking with its erase-on-allocate strategy.

**Holder:** A user who possesses a particular identifier. The term holder is used in conjunction with the term identifier. Users are said to be the holders of identifiers if they possess the identifiers. The rights database is the place in the system where the associations of users and the identifiers they hold are permanently kept. However, each process also has a right list that includes all the identifiers the process is authorized to hold.

## Glossary

**Identifier:** A notation that defines a user or group of users. There are three types of identifiers: UIC identifiers, system-defined identifiers, and general identifiers.

**Locked Password:** A password that cannot be changed by the account's owner. Only system managers or users with the SYSPRV privilege can change locked passwords.

**Login:** The series of actions involved in authenticating a user to the system and creating a process that runs on the user's behalf.

**Mandatory Controls:** Security controls that are imposed by the system upon all users. There are no examples of mandatory controls within the VAX/VMS system. Mandatory controls are the opposite of discretionary controls. Thus, they are synonymous with the term nondiscretionary controls.

**Nondiscretionary Controls:** *Mandatory Controls*

**Open Accounts:** Accounts that do not require passwords.

**Passwords:** Character strings that users provide at login time to validate their identity and as a form of proof of their authorization to access the account. There are two kinds of passwords, system passwords and user passwords. User passwords include both primary and secondary passwords.

**Primary Password:** A type of user password that is the first user password requested from the user. Systems may optionally require a secondary password, as well. As a user password, this password must be the password that is associated with the user name that is supplied with it.

**Privileges:** A means of protecting the use of certain system functions that can affect system resources and/or integrity. System managers grant privileges according to user's needs and deny them to users as a means of restricting their access to the system.

**Proxy Login:** A type of login that permits a user from a remote node to effectively login to a local node, as if the user owned an account on the local node. However, the user does not specify a password in the access control string. In some cases, the remote user does own the account, but in other cases the remote user shares the account with other users.

**Rights Database:** The collection of data the system maintains and uses to associate identifiers and the holders of the identifiers with their rights, attributes, and so forth.

**Rights List:** The list associated with each process that includes all the identifiers the process holds.

**Secondary Password:** A user password that may be required at login time, immediately after the primary password has been correctly submitted. Primary and secondary passwords can be known by separate users, to ensure that more than one user is present at the login. A less common use is to require a secondary password as a means of increasing the password length so that the total number of combinations of characters makes password more time-consuming.

**Secure Terminal Server:** A piece of VAX/VMS software designed to ensure that users can only login to terminals that are already logged out. When the user presses the BREAK key on a terminal, the secure server (if enabled) responds by first disconnecting any logged in process and then initiating a login. If no process is logged in at the terminal, the login can proceed immediately.

**Security Alarm:** A message sent to operator terminals that are enabled as security operators. Security alarms are triggered by the occurrence of an event previously designated as worthy of the alarm because of its security implications.

**Security Manager:** In the VAX/VMS context, the person or persons responsible for protecting the security of the computer system. This role is sometimes performed by the same person or persons who function as system managers. It requires the same skills as the system manager, but additional privilege (the SECURITY privilege) as well as knowledge of the security features provided with the VAX/VMS operating system.

**Security Operator Terminal:** A class of terminal that has been enabled to receive messages sent by OPCOM to "security operators". These messages are security alarm messages. Normally such a terminal is a hardcopy terminal in a protected room, so that the output provides a log of security-related events and details that identify the source of the event.

**System-Defined Identifier:** One of three classes of identifiers.

System-defined identifiers are provided by the system to identify groups of users according to their usage of the system. For example, all users who access the system by dialing up receive the DIALUP identifier.

**System Password:** A password required by a terminal before login can be initiated at the terminal.

**Tied Account:** *Captive Account*

**Trojan Horse Program:** A program that gains access to otherwise secured areas through its pretext of serving one purpose when its real intent is far more devious and potentially damaging.

**Turnkey Account:** *Captive Account*

**UIC:** *User Identification Code*

**User Identification Code:** A coded notation that represents a user of the system. Normally each user has a unique user identification code, although at a few sites some users may share the same UIC. In that case, there is no way for the system to distinguish one user from another. User identification codes include a designation of the user and the user's group.

**User Password:** A password that is associated with a user. This password must be correctly supplied when the user attempts to login so that the user is authenticated for access to the system. The two types of user passwords are known as primary and secondary, to suggest the sequence in which they are entered.

**Worm-hole:** A flaw in the file protection that allows a "worm" (or procedure) to access another command procedure or executable image and modify that file so that it carries a copy of the worm. Each time an unsuspecting user executes the code that contains the worm, the worm attempts to propagate itself into other poorly protected procedures or images, traveling along the worm-hole. The worm seeks to find its way into a procedure that will be run from a privileged account so that the worm can inflict damage to the system security.

# Index

## A

- Access
  - causes security alarm • 4-25
  - CONTROL • 4-8
  - DELETE • 4-7
  - denying to class of users
    - example • 5-9
  - denying with identifier ACE • 4-34
  - EXECUTE • 4-7
  - file
    - denying through protection code • 4-9
  - granting or denying through ACE • 4-28
  - how system determines introduction to • 4-2
  - logical I/O
    - to magnetic tape • 4-15
  - physical I/O
    - to magnetic tape • 4-15
  - READ • 4-7
  - to disk file • 4-11
  - UIC-based protection code effects • 4-6
  - WRITE • 4-7
- Access category
  - group • 4-6
  - owner • 4-6
  - system • 4-6
  - world • 4-6
- Access Control List
  - See ACL
- Access Control List Entry
  - See ACE
- Access control string • 3-34
  - cannot accomodate secondary password • 5-29
  - revelation of password • 3-25
- Access matrix • 4-17 to 4-20
- /ACCESS qualifier
  - AUTHORIZE • 5-48
- Access request
  - to file
    - details of evaluation • 4-43
- Access type
  - abbreviation • 4-8
  - and security audit • 4-54
  - meaning for directory file • 4-12
  - meaning for disk file • 4-11
  - meaning for volume • 4-14
- Access, proxy
  - conditions where nonfunctional • 7-19
  - requirements • 7-19
- Account
  - captive • 3-13
    - disabling mail and notification of delivery • 5-37
    - for network environment • 7-8
  - DECNET
    - example • 7-10
  - disable
    - with /FLAGS=DISUSER • 5-49
  - duration
    - how to set • 5-50
  - emergency
    - and privileges • 5-57
  - FAL
    - example • 7-10
    - where appropriate • 7-13
  - guest
    - why not recommended • 5-83
  - identity
    - disguising • 6-6
  - locked password • 3-13
  - network
    - guidelines for establishment • 7-7
  - network default
    - considerations for privileges • 7-7
  - open • 3-13
  - privileged
    - restriction suggestions • 5-58
  - proxy • 3-35
    - example • 7-24

## Index

- Account (cont'd.)
  - setting up to use project identifiers • 5-22
  - supersensitive
    - use of dual passwords • 3-22
  - user
    - guidelines for establishment • 5-2 to 5-67
- Account expiration • 3-28
- Account lifetime • 3-28
- Account, proxy
  - as captive account • 7-16
  - example • 7-16
  - recommended restrictions • 7-15
- Accounting log
  - as security tool • 6-5
- Accounts
  - multiple
    - and passwords • 3-27
- ACE • 4-24
  - automatically added
    - for file creation by non-owner • 4-42
  - default
    - specified by DEFAULT option • 4-30
  - default protection • 4-25, 5-14
    - example • 5-19, 7-25
  - default protection type • 4-30
  - deletion of • 4-27
  - identifier • 4-25
  - options field • 4-27
  - positioning considerations • 4-24, 4-29
    - rules • 4-34
  - propagation
    - suppressed with NOPROPA-GATE • 4-27
  - protecting from accidental deletion • 4-27
  - security alarm • 4-25, 4-31
  - syntax • 4-26 to 4-33
  - types • 4-25
- ACL • 4-17 to 4-35
  - advantage of shortness • 5-8
  - commands
    - use of wildcards in • 4-42
  - creation and maintenance • 4-20
  - disadvantages • 5-8
- ACL (cont'd.)
  - maintaining current
    - hint • 4-35
  - management
    - to avoid pitfalls • 4-34
  - usage considerations • 4-34
  - use for file sharing over network • 7-21
  - use on system program files • 5-50
- ACL Editor • 4-20
  - example • 5-12
- ACLs
  - introduction to • 4-1
- ACNT privilege • A-1
- ADD/IDENTIFIER AUTHORIZE
  - command • 5-10, 5-22
- ADD/PROXY AUTHORIZE command • 7-16, 7-22
- Alarm
  - ACE • 4-25
    - for failed file access attempt • 4-33
    - for successful file access • 4-33
  - security
    - applications • 4-54
- ALARM\_JOURNAL
  - specifies alarm ACE • 4-32
- ALF • 5-43 to 5-46
  - maintenance recommendations • 5-44
- ALFMAINT command procedure • 5-44
  - See ALF
- ALFMAINT.COM • 5-43
- Algorithm
  - password encryption • 3-10
- ALLSPOOL privilege • A-1
- Alphanumeric UIC • 4-4
- ALTPRI privilege • A-2
- Announcement message • 3-6
  - security disadvantage • 5-35
- APPEND/PROTECTION command • 5-19
- Attack
  - security
    - forms of • 6-1
- Attribute
  - resource • 4-35
- Attribute, resource • 5-20
  - example • 5-20

## Index

Audit trail  
  in security model • 2-2  
  role in security • 2-6  
Auditing  
  applications • 6-6  
  as security feature • 6-6  
  security • 4-52  
Authenticating  
  users • 3-9  
    through visual verification • 3-22  
Authorization Database  
  role in security • 2-5  
Authorization database  
  concept • 4-17  
  considerations on a VAXcluster •  
    8-2  
  defined • 2-2  
Autoanswer  
  and backup synchronous dialup • 7-9  
Autobauding • 3-12  
Autologin  
  account  
    as security problem • 5-46  
  See ALF • 5-43  
Autologin file  
  VAXcluster requirements • 8-2  
Automatic  
  login file  
    See ALF  
Automatic generation  
  of password • 3-15  
Automatic password generator  
  use to obtain initial password • 5-24  
  when to require • 5-32

---

## B

Backup operations  
  general recommendations • 5-74  
  performed as captive privileged  
    account • 5-59  
BATCH  
  as system identifier • 4-22  
  identifier • 5-9  
Batch jobs  
  affected by shift restrictions • 3-31  
Batch login • 3-4  
Baud rate  
  automatic setting of • 3-12

Binary file  
  not appropriate for MAIL transfer •  
    7-20

BOTH  
  NCP parameter • 7-18  
Brackets  
  use in UICs • 4-5  
Breach  
  handling • 6-7  
BREAK key  
  and secure server • 5-42  
Breakin  
  attempts  
    and security audit • 4-54  
  counteraction through dual  
    password • 5-29  
  detection and evasion  
    controlling • 5-38 to 5-41  
    loopholes • 5-41  
  evasion • 3-32  
Breakin attempt  
  dialup type • 3-32  
Breakin evasion  
  as cause of login failures • 5-40  
Browser  
  catching  
    with alarm • 4-55  
  identification of • 6-8  
Browsers  
  handling the problem • 6-8  
  how to catch • 6-6  
  tricks of • 3-25  
BUGCHK privilege • A-2  
BYPASS privilege • 4-9, A-3  
  affect on ownership privilege • 4-37

---

## C

Captive  
  account  
    defined • 3-13  
Captive account • 5-79 to 5-86  
  and locked password • 5-80  
  as target for penetrators • 5-79  
  creation • 5-79  
  danger of process spawning • 5-79  
  disabling mail and notification of  
    delivery • 5-37

## Index

Captive account (cont'd.)  
  example of production account • 5-61  
  for network environment • 7-8  
  special case  
    allowed privileges • 5-59  
Catching probers • 6-6  
Catching probing outsiders • 6-6  
Categories  
  of user • 4-1  
Changing  
  UIC-based protection • 4-14 to 4-16  
Characteristics  
  NCP display • 7-18  
Charging  
  disk space usage  
  by identifiers • 4-35  
Circuit  
  verification • 7-9  
Circuit database  
  guidelines • 7-9  
Clearing  
  screen of video terminal • 3-43  
/CLITABLES qualifier  
  AUTHORIZE • 5-50, 5-80  
Cluster  
  See VAXcluster  
Cluster manager  
  and security manager • 8-1  
CMEXEC privilege • A-3  
CMKRNL privilege • A-4  
Collision  
  password • 5-25  
Command  
  usage restrictions • 5-50  
Common database  
  authorization  
    benefits for VAXcluster security • 8-2  
Compiler  
  restricting use with ACLs • 5-70  
Conduit  
  application for network security • 7-6  
CONNECT/CONTINUE command • 3-44  
CONTROL access • 4-8  
  and FAL account • 7-8  
  and READALL privilege • 4-10  
  changing directory protection • 4-15

CONTROL access (cont'd.)  
  meaning for directory file • 4-12  
  meaning for disk file • 4-11  
  meaning for volume • 4-14  
  meaning in ACL • 4-28  
COPY/PROTECTION command • 5-19  
Costs  
  of security • 1-10  
CREATE DIRECTORY/PROTECTION  
  command • 4-40  
CREATE/DIRECTORY command  
  and DELETE access • 4-13  
CREATE/DIRECTORY/OWNER\_UIC  
  command • 4-38  
CREATE/PROXY AUTHORIZE  
  command • 7-15  
CREATE/RIGHTS AUTHORIZE  
  command • 5-9  
  example • 5-10  
CTRL/B command • 3-41  
CTRL/Y  
  and captive accounts • 5-79, 5-82  
CTRL/Z  
  use with ACL Editor • 5-12

---

## D

Data Security Erase  
  See DSE  
Database  
  authorization  
    concept • 4-17  
    considerations on a VAXcluster • 8-2  
  DECnet node and circuit  
    guidelines • 7-9  
  rights • 4-17  
    creating and maintaining • 5-9, 5-14  
    introduction • 4-5  
Database, common  
  authorization  
    benefits for VAXcluster security • 8-2  
Database, rights  
  display • 5-13  
Date  
  expiration  
    account • 3-29



## Index

- DCL tables
  - modifications for security • 5-50
- Debugging enabled
  - security hazard • 5-56
- DECnet
  - and VAXcluster nodes • 8-4
- DECNET account
  - example • 7-10
- Default
  - file owner
    - how established • 4-39
  - protection • 4-40
- Default ACE
  - specified by DEFAULT option • 4-30
- Default network account
  - and reference monitor • 7-5
- Default ownership
  - management • 5-14 to 5-18, 5-19 to 5-23
- Default process protection • 4-41, 5-14, 5-18
- Default protection
  - directory file
    - specified by ACE • 4-27
  - for directories
    - role of MFD • 4-15
  - management • 5-14 to 5-19
  - propagating with ACE • 4-30
- Default protection ACE
  - adding or changing • 4-31
- DEFAULT\_PROTECTION ACE • 4-30
- Defaults
  - protection • 4-43
- DELETE access • 4-7
  - meaning for directory file • 4-12
  - meaning for disk file • 4-11
  - meaning for volume • 4-14
  - meaning in ACL • 4-28
- DELETE/ERASE command • 4-49
- Denial of service problem
  - potentially induced through evasive action • 5-41
- Departments
  - role in group design • 5-3
- Design
  - of identifiers • 5-7
- DETACH privilege • A-4
- Detached process login • 3-5
- Detecting actual account being abused
  - 6-6
- Detection
  - of breakin attempts • 3-32
- Device
  - restrictions for users • 5-46
- Device protection
  - through identifier ACEs • 4-30
- Devour privileges • 5-53
- DIAGNOSE privilege • A-4
- DIALUP
  - as system identifier • 4-22
  - identifier • 5-9
- Dialup
  - backup synchronous
    - and autoanswer • 7-9
  - retries
    - controlling • 5-37
- Dialup connection
  - breaking properly • 3-45
- Dialup login • 3-3
  - failures • 3-32
- Directory
  - access
    - execute-only • 4-12
  - deleting file • 4-13
  - owner
    - default assignment • 4-38
  - ownership
    - how established • 4-38
  - propagating identifier ACE
    - through DEFAULT option • 4-29
- Directory file
  - default protection • 4-40
    - ACL-based • 4-40
    - UIC-based • 4-40
- DIRECTORY/OWNER command
  - use to display directory owner • 4-38
- DIRECTORY/SECURITY command • 4-50
- Disconnected job message • 3-7
- Disconnected Jobs
  - feature
    - management • 5-35
- Disconnected processes
  - at logout time • 3-44
- Disk
  - default protection • 4-41
  - file access • 4-11

## Index

Disk (cont'd.)  
  protection • 4-3  
Disk quota  
  as restriction for user • 5-47  
  example • 5-22  
Disk scavenging • 4-48  
  how to discourage • 5-75  
Disk space  
  usage and charging • 5-20  
Disk space usage  
  charging to identifiers • 4-35  
Disk volume  
  restrictions • 5-47  
DSE • 5-75, 5-76  
  and erasure pattern • 4-49  
  use of random pattern  
  advantages • 5-76  
Dual password  
  advantages and disadvantages •  
  5-28  
Dual passwords  
  and maximum security • 5-23  
Duration  
  account • 3-29  
  how to set • 5-50  
  evasive action • 5-40

---

**E**

---

Editor  
  ACL • 4-20  
  example • 5-12  
  used to delete ACE • 4-27  
  caution for use in captive command  
  procedure • 5-83  
Emergency account  
  and privileges • 5-57  
Encoding  
  password • 3-11  
Encryption  
  one-way  
    and collision • 5-25  
  password • 3-11  
Encryption algorithm  
  one-way  
    for passwords • 3-10  
Entry  
  illegal • 3-32

Environmental factors  
  in security • 1-5  
Erase-on-allocate • 4-50, 5-77  
Erase-on-delete  
  security feature • 5-76  
Erasure pattern • 4-49, 5-75  
Establishing  
  ACLs • 4-20  
  UIC-based protection • 4-14 to 4-16  
Ethernet  
  lack of protection • 7-6  
Evasion  
  breakin • 3-32  
Evasive action  
  duration • 5-40  
  invoked as counteraction for breakin  
  • 5-38  
Exchanging  
  files  
    in network environment • 7-20  
EXECUTE access • 4-7  
  meaning for directory file • 4-12  
  meaning for disk file • 4-11  
  meaning for volume • 4-14  
  meaning in ACL • 4-28  
Expiration  
  account • 3-28  
  password • 3-19  
    how to pre-expire • 5-25  
    how to set • 5-29  
    not applicable to system  
    password • 5-27  
/EXPIRATION qualifier  
  AUTHORIZE • 5-50  
EXQUOTA privilege • A-5  
External node  
  and default access rights • 7-9

---

## F

---

Failures  
  login  
    causes of • 3-29  
    how counted for breakin  
    detection • 5-39  
FAL account • 7-8  
  and CONTROL access • 7-8  
  example • 7-10  
  where appropriate • 7-13

## Index

- False alarms
    - proper handling • 6-3
  - Fiber optics
    - application for network security • 7-6
  - File
    - creation
      - flowchart • 5-15
    - protection • 4-3
    - sensitive
      - application of alarm • 4-55
    - sharing and exchanging
      - in network environment • 7-20 to 7-27
    - transfers
      - with MAIL • 7-20
    - write-only
      - not supported by VAX/VMS • 4-11
  - File access
    - See also UIC
    - category summary • 4-6
    - CONTROL • 4-8
    - DELETE • 4-7
    - EXECUTE • 4-7
    - how system evaluates • 4-2
    - READ • 4-7
    - WRITE • 4-7
  - File access request
    - how evaluated
      - in detail • 4-43
  - File owner
    - how established by default • 4-39
  - File protection • 4-1 to 4-57
    - and RENAME command • 4-43
    - and system security • 4-1
    - changing • 4-16
    - changing default • 4-41
    - default ACL-based • 4-42
    - default disk • 4-41
    - default UIC-based • 4-41
    - establishing and changing • 4-16
    - faulty invites worms • 5-73
    - of magnetic tape volumes • 4-14
    - violations
      - auditing • 6-6
  - File sharing
    - considerations for a VAXcluster • 8-3
  - Files privileges • 5-54
  - Files-11 structure
    - non-hierarchical nature • 4-12
  - /FLAGS=CAPTIVE qualifier
    - AUTHORIZE • 5-79
  - /FLAGS=DISMAIL qualifier
    - AUTHORIZE
      - use with DISNEWMAIL flag • 5-37
  - /FLAGS=DISNEWMAIL qualifier
    - AUTHORIZE
      - use to suppress notification • 5-37
  - /FLAGS=DISRECONNECT qualifier
    - AUTHORIZE • 5-36
  - /FLAGS=DISREPORT qualifier
    - AUTHORIZE
      - use to disable last login messages • 5-36
  - /FLAGS=DISUSER qualifier
    - AUTHORIZE • 5-33
  - /FLAGS=DISWELCOME qualifier
    - AUTHORIZE
      - use to disable welcome message • 5-36
  - /FLAGS=GENPWD qualifier
    - AUTHORIZE • 5-32
      - to invoke password generator • 5-29
  - /FLAGS=LOCKPWD qualifier
    - AUTHORIZE
      - use to control passwords • 5-32
  - /FLAGS=PWD\_EXPIRED qualifier
    - AUTHORIZE
      - use to give second chance • 5-30
  - Foreign countries
    - and network usage restrictions • 7-10
  - Forgery
    - of network information • 7-6
  - Format
    - UIC • 4-4
  - Fragmented disks
    - impact on highwater marking • 5-77
- 
- ## G
- 
- General identifier • 4-21, 4-22

## Index

General identifier (cont'd.)  
    deleted  
        how to recognize • 5-12  
        reasons for using • 4-34  
/GENERATE\_PASSWORD  
    AUTHORIZE qualifier • 5-24  
Generator  
    automatic password  
        for initial password • 5-24  
Goals  
    of security managers • 1-1  
Grabber  
    password  
        symptom and counteraction • 4-53  
Grabbers  
    password • 3-23  
GRANT/IDENTIFIER AUTHORIZE  
    command • 5-10, 5-22  
GROUP  
    user category • 4-6  
Group  
    design of • 5-3 to 5-14  
    impact on user privileges • 5-3  
    name  
        in UIC • 4-5  
    number  
        in UIC • 4-4  
    number of per member • 4-5  
    overlapping user • 4-17  
    user  
        defined by holders of identifiers • 4-17  
Group number  
    uniqueness requirement  
        for VAXcluster • 8-3  
GROUP privilege • 5-53, A-5  
GRPNAM privilege • A-5  
GRPPRV  
    and user category • 4-6  
GRPPRV Privilege • A-6  
GRPPRV privilege • 4-9  
    affect on ownership privilege • 4-37  
Guest account  
    why not recommended • 5-83  
Guest accounts  
    improving security  
        as captive accounts • 5-83

---

## H

---

Hacker  
    as security problem • 1-5  
Hardcopy terminal  
    logout considerations • 3-44  
Heterogeneous VAXcluster  
    See Nonhomogeneous VAXcluster  
Hexadecimal  
    identifier  
        in ACE • 5-12  
Hexadecimal format  
    UIC identifier • 4-22  
Highwater marking • 4-49, 5-77  
    and performance • 5-77  
Holder  
    displaying records • 5-14  
    how to associate with identifier • 5-10  
    removal • 5-11  
Homogeneous VAXcluster • 8-2

---

Identification  
    versus prevention • 6-11  
Identifier  
    associating with holders • 5-10  
    combined in one ACE  
        example • 5-9  
    design considerations • 5-7  
    general • 4-21, 4-22  
    groups by areas of interest • 4-19  
    hexadecimal format in ACE • 5-12  
    removal • 5-11  
    reserved  
        See Identifier, system-defined  
    sharing same • 4-26  
    specifying multiple in ACE • 4-26  
    system-defined • 4-22, 4-23  
    types • 4-21  
    uniqueness requirement  
        for VAXcluster • 8-3  
    use of wildcards in for ACE  
        example • 4-29  
Identifier ACE  
    syntax • 4-26

## Index

Identity  
  disguised  
    catching before • 6-6

Illegal  
  entry • 3-32

Image  
  installing with privilege  
    security ramifications • 5-56

INCOMING  
  NCP parameter • 7-18

Indicators  
  of trouble • 6-2

INITIALIZE/ERASE command • 5-76

INQUIRE command  
  reasons to omit from captive  
    command procedures • 5-82

Installing image  
  with privilege  
    security ramifications • 5-56

INTERACTIVE  
  as system identifier • 4-22  
  identifier • 5-9

Interactive login  
  defined • 3-2  
  vs interactive mode process • 3-3

---

## J

Job controller  
  affected by shift restrictions • 3-31  
  enforces work time restrictions •  
    5-48  
  impact of SET DAY command on  
    logged in processes • 5-48

Job termination  
  imposed by shift restrictions • 3-31

Jobs  
  batch  
    affected by shift restrictions •  
      3-31

---

## K

Kernel  
  security • 2-3

---

## L

Last login messages • 3-8  
  disabling with /FLAGS=DISREPORT •  
    5-36  
  using • 4-52

LAT-11  
  considerations for breakin detection  
    • 5-39  
  incompatible with system password  
    • 5-26

Levels of security  
  defined • 1-7

LGI  
  SYSGEN parameters • 5-37

LGI\_BRK\_DISUSER  
  SYSGEN parameter • 5-41

LGI\_BRK\_LIM  
  SYSGEN parameter • 5-38

LGI\_BRK\_TERM  
  SYSGEN parameter • 5-39

LGI\_BRK\_TMO  
  SYSGEN parameter • 5-39

LGI\_HID\_TIM  
  SYSGEN parameter • 5-40

LGI\_RETRY\_LIM  
  SYSGEN parameter  
    use to control retries for dialup •  
      5-37

LGI\_RETRY\_TMO  
  SYSGEN parameter  
    use to control retries for dialup •  
      5-37

/LGICMD qualifier  
  AUTHORIZE  
    and captive accounts • 5-81

Lifetime  
  account • 3-28  
  password • 3-18

LINK/NOTRACE command  
  recommended for images installed  
    with privilege • 5-56

List  
  rights • 4-23

LOCAL  
  as system identifier • 4-22  
  identifier • 5-9

## Index

- Local area networks
    - lack of protection • 7-6
  - Local login • 3-3
  - Locked password
    - account • 3-13
    - advantage • 5-32
  - Locking up
    - terminal areas • 3-41
  - LOCKPWD
    - flag
      - in AUTHORIZE • 3-13
  - Logging in
    - See login
  - Logging out
    - after remote logins • 3-44
    - security considerations • 3-39 to 3-45
    - with disconnected processes • 3-44
  - Login • 3-1 to 3-34
    - and default process protection • 4-41
    - batch • 3-4
    - class • 3-2
      - restrictions • 3-31
    - controlling number of attempts
      - for dialup • 5-37
    - denied for expired accounts • 3-29
    - detached process • 3-5
    - dialup • 3-3
      - chances to supply password • 3-32
    - disabled
      - by breakin evasion • 3-32
      - by shift restriction • 3-31
    - failures • 3-8
      - and retries • 3-32
      - counting for breakin detection • 5-39
    - interactive
      - defined • 3-2
    - last chance due to password expiration • 3-20
    - local • 3-3
    - message • 3-5
      - suppression • 3-9
    - network • 3-4
    - noninteractive
      - defined • 3-2
    - permitted time periods • 3-31
  - Login (cont'd.)
    - proxy • 3-4
      - and the user • 3-35
      - establishment and management • 7-14 to 7-20
    - remote • 3-3
      - and system password • 5-27
    - simplifying for user
      - with ALF • 5-45
    - subprocess • 3-5
    - time out • 3-22
    - type
      - as system identifier • 4-22
  - Login command procedure
    - command to deny remote file access • 7-8
    - proper protection for • 5-74
  - Login failures
    - causes • 3-29 to 3-34
  - Login message
    - controlling • 5-35 to 5-37
  - Login program
    - and rights list • 4-23
    - authentication by secure server • 3-23
  - LOGIO privilege • A-6
  - LOGOUT command • 3-43
  - LOGOUT/HANGUP command • 3-45
  - Longevity
    - evasive action • 5-40
- 
- ## M
- 
- Magnetic tape
    - EXECUTE and DELETE access
      - invalid for volumes • 4-14
    - foreign
      - access to • 4-15
      - protection • 4-3, 4-14
      - volume
        - protection code • 4-9
  - MAIL
    - and system security • 3-38
  - Mail
    - notification message
      - controlling • 5-37
  - MAIL command
    - used to transfer text files • 7-20

## Index

- Mail file
  - recommended protection • 4-51
- Marking
  - highwater • 4-49
- Mask
  - protection
    - specifying in ACE • 4-30
- Master file directory
  - See MFD
- Matrix
  - access • 4-17 to 4-20
- MAXSYSGROUP
  - and SYSTEM category • 4-6
- Media initialization
  - restricting with ACLs • 5-70
- Member
  - name
    - in UIC • 4-5
  - names
    - uniqueness requirement • 4-5
  - number
    - in UIC • 4-4
- Memory consumption
  - paged system dynamic and ACLs • 5-8
- Message
  - announcement • 3-6
  - disconnected job • 3-7
  - last login • 3-8
  - welcome • 3-7
- Messages
  - last login
    - disabling with /FLAGS=DISREPORT • 5-36
  - login • 3-5
- MFD
  - provides default protection • 4-15
- MODIFY/SYSTEM\_PASSWORD
  - AUTHORIZE command • 5-27
- Monitor protection
  - with DIRECTORY/SECURITY command • 4-50
- Mount
  - volume
    - and security audit • 4-54
- MOUNT privilege • A-7

---

## N

---

- Name
  - group
    - in UIC • 4-5
  - member
    - in UIC • 4-5
- Names
  - use as passwords • 3-14
- NCP
  - using to enhance network security • 7-17
- NETMBX privilege • A-7
- NETUAF • 3-35
- NETUAF.DAT
  - and wildcards • 7-22
  - normal protection • 5-34
  - proxy login file
    - automatic maintenance • 7-16
- NETWORK
  - as system identifier • 4-22
  - identifier • 5-9
- Network
  - encryption
    - lack of • 7-6
  - protected communications
    - security problem • 7-6
  - security • 7-1 to 7-27
    - limitations • 7-1
  - usage restrictions
    - in foreign countries • 7-10
- Network access control string
  - cannot accomodate secondary password • 5-29
  - revelation of password • 3-25
- Network accounts
  - guidelines for establishment • 7-7
- Network Control Program
  - See NCP
- Network default account
  - and WORLD access • 7-6
- Network login • 3-4
- Network password
  - guidelines • 7-9
- Network security
  - user considerations for • 3-34
- Network User Authorization File
  - See NETUAF

## Index

Networks  
  security problems of • 1-6  
Node  
  external  
    and default access rights • 7-9  
Node database  
  guidelines • 7-9  
Node name  
  revealed at logout • 3-43  
NONE  
  NCP parameter • 7-18  
Nonhomogeneous VAXcluster • 8-2  
Noninteractive login  
  defined • 3-2  
NORESOURCE  
  attribute • 4-35  
Normal privilege • 5-53  
Number  
  group  
    in UIC • 4-4  
  member  
    in UIC • 4-4  
Numeric UIC • 4-4

---

## O

Objects  
  in security model • 2-2  
  role in security • 2-5  
One-way encryption  
  and collision • 5-25  
Online debugging  
  as security hazard • 5-56  
Open account • 3-13  
  and captive account • 5-80  
  captive recommendation • 5-34  
Open files  
  and ACL consumption of memory • 5-8  
OPER privilege • A-7  
Operator  
  group assignment • 4-6  
  terminal  
    enabled for security alarms • 4-31  
Options field  
  in identifier ACE • 4-27  
Order  
  of ACEs • 4-24, 4-34

Order  
  of ACEs (cont'd.)  
    recommendations and example • 4-29  
OUTGOING  
  NCP parameter • 7-18  
Outsiders  
  catching probing • 6-6  
OWNER  
  user category • 4-6  
    access to magnetic tape • 4-9  
Owner  
  changing identifier of  
    resource attribute example • 4-39  
  default  
    for directory • 4-38  
Ownership  
  directory  
    how established • 4-38  
  effects on protection checks • 4-35  
  establishing and changing • 4-35 to 4-39  
  how assigned during file creation • 5-15  
  privileges  
    how obtained • 4-36  
Ownership defaults  
  management • 5-14 to 5-18, 5-19 to 5-23

---

## P

Password  
  automatic generation of new • 3-15  
  chances to supply during dialups • 3-32  
  change frequency guidelines • 3-27  
  changes  
    number of • 3-18  
  changing • 3-15  
  choices  
    bad • 3-14  
    good • 3-21  
  dual • 3-22  
    use of • 5-23  
  elimination  
    for networks • 7-20  
  encoding • 2-4  
  encryption • 3-10



## Index

### Password (cont'd.)

- English word
  - consequence of • 3-16
- expiration • 3-19
  - how to pre-expire • 5-25
  - how to set • 5-29
- first name as • 3-14
- generator
  - use to obtain initial password • 5-24
- guessing
  - antidote • 5-40
  - counteractions • 3-16
  - handling • 6-8
  - identification of source • 6-8
  - prevention • 6-9
- guessing and length • 3-20
- how to pre-expire • 5-25
- hunters
  - and dialup retries • 3-32
  - tricks of • 3-26
- initial • 5-24
- keeping old • 3-20
- length
  - minimum • 5-31
- lifetime • 3-18
- locked • 3-13
  - advantage • 5-32
  - for captive accounts • 5-80
- management • 5-23 to 5-34
- minimum length • 3-14, 3-20
- and automatic generation • 3-16
- network
  - guidelines • 7-9
- null
  - as choice for captive account • 5-80
- pre-expired
  - how to recognize • 5-25
- primary • 3-21, 5-24
- protection
  - guidelines • 5-33
- retries • 3-32
- role in security • 2-4
- secondary • 3-21, 5-28
- secrecy • 3-9
  - and electronic mail • 3-27
- secure choice for • 3-21
- selection • 3-14
- selection guidelines • 3-21

### Password (cont'd.)

- sharing • 3-28
  - poor practice • 7-21
- storage • 3-10
- system • 3-12
  - See also System password
  - as cause of login failures • 3-30
  - guidelines • 5-27
- use on multiple systems • 3-27
- user
  - defined • 3-10
  - uniqueness on each account • 3-27
  - user selection of new • 3-15
  - verification of new during change • 3-15
- Password collision • 5-25
- Password encryption
  - and collision • 5-25
- Password grabber
  - and logouts • 3-43
  - secure server
    - as antidote • 5-42
  - symptom and counteraction • 4-53
- Password grabbers • 3-23
- Password protection • 3-25
- /PASSWORD qualifier
- AUTHORIZE
  - to specify secondary password • 5-29
- Password stealing programs • 3-23
- Password, system • 5-26 to 5-28
- Passwords
  - user's perspective of • 3-9 to 3-28
- Pattern
  - erasure • 4-49
- Penetration
  - as security problem • 1-4
- Penetrator • 1-5
- Performance
  - and ACL length • 5-8
  - and ACLs • 4-34
  - and automatic password generator • 5-31
  - and highwater marking • 5-77
  - and security • 1-10
  - effect of poor security on • 3-46
- PFNMAP privilege • A-8
- PHY\_IO privilege • A-8

## Index

- Physical security • 1-5
  - of networks • 7-6
- Ports
  - publicly accessible
  - defending • 5-28
- /PRCLM qualifier
  - AUTHORIZE
    - use to prohibit spawning in captive accounts • 5-79
- Pre-expired password
  - how to set and observe • 5-25
- Prevention
  - versus identification • 6-11
- Primary day
  - how to define • 5-48
- /PRIMEDAYS qualifier
  - AUTHORIZE
    - example • 5-48
- Privilege
  - all • 5-54
  - alternatives to • 5-54
  - BYPASS • 4-9
  - disabling • 5-54
  - for captive account
    - special case • 5-59
  - group-related • 5-3
  - ownership
    - how obtained • 4-36
  - process
    - granting • 5-51 to 5-59
  - recommendations for minimum • 5-57
  - requirements for security manager • 5-2
  - summary • 5-52
  - use of to gain access and security audit • 4-54
  - user
    - how defined • 5-47
- Privilege vector • 5-52
- Privileged account
  - considerations for network • 7-7
- Privileges
  - how acquired by outsiders • 5-55
  - used for file sharing
    - poor practice • 7-21
- /PRIVILEGES qualifier
  - AUTHORIZE
    - use to define user privileges • 5-47
- PRMCEB privilege • A-9
- PRMGBL privilege • A-9
- PRMMBX privilege • A-10
- Prober
  - outsider
    - counteraction • 5-38
- Probers
  - how to catch • 6-6
- Probing
  - as security problem • 1-3
- Process
  - rights list • 4-23
- Process privilege • 5-51
- Process protection
  - default • 4-41
- Process protection, default • 5-14, 5-18
- Process reconnection • 3-7
- Programs
  - for stealing passwords • 3-23
- Prohibitions
  - for login class • 3-31
  - shift • 3-31
- Prompt
  - username
    - use ALF to suppress • 5-45
- Propagation
  - ACE
    - use NOPROPAGATE to suppress • 4-27
  - of protection • 4-40 to 4-43
  - in directories • 4-25
  - protection
    - example • 7-25
- Protection • 4-3
  - access category • 4-6
  - and RENAME command • 4-43
  - bypassing checks • 4-9
  - changing • 4-16
  - changing default • 4-41
  - code
    - how assigned during file creation • 5-15
  - default • 4-40
  - role of MFD for directories • 4-15

## Index

Protection (cont'd.)  
  default for files in directory  
    specified by ACE • 4-27  
  defaults  
    management • 5-14 to 5-19  
  device  
    through identifier ACEs • 4-30  
  file  
    default UIC-based • 4-41  
    faulty invites worms • 5-73  
  of command procedures  
    as antidote for worms • 5-74  
  of directories • 4-12  
  of files • 4-3  
  of magnetic tape files • 4-14  
  of magnetic tape volumes • 4-14  
  of password • 3-25  
  of volumes • 4-3  
  propagation of • 4-40 to 4-43  
  specification of • 4-8, 4-9  
  UIC-based • 4-3  
Protection check  
  influenced by ownership • 5-15  
Protection checking  
  UIC-based • 4-6  
Protection defaults • 4-43  
Protection mask  
  specifying in ACE • 4-30  
Protection, default process • 5-14,  
  5-18  
Protection, file • 4-1 to 4-57  
Protection, UIC-based • 4-1 to 4-16  
Proxy access  
  conditions where nonfunctional •  
    7-19  
  requirements • 7-19  
Proxy Account  
  as captive account • 5-86  
Proxy account • 3-35  
  as captive account • 7-16  
  example • 7-16, 7-24  
  for multiple users • 3-37  
  for single user • 3-37  
  recommended restrictions • 7-15  
Proxy accounts  
  and VAXclusters • 8-4  
Proxy login • 3-4  
  and circuit verification • 7-9  
  and the user • 3-35

Proxy login (cont'd.)  
  establishment and management •  
    7-14 to 7-20  
  key characteristic • 3-38  
PSWAPM privilege • A-10  
Publicly accessible ports  
  defending • 5-28  
PURGE/ERASE command • 4-49  
/PWDLIFETIME qualifier  
  AUTHORIZE  
    controls password expiration •  
      5-29  
/PWDMINIMUM qualifier  
  AUTHORIZE • 5-31

---

## Q

Quota  
  disk  
    charging to identifiers • 4-35

---

## R

READ access • 4-7  
  and READALL privilege • 4-10  
  meaning for directory file • 4-12  
  meaning for disk file • 4-11  
  meaning for volume • 4-14  
  meaning in ACL • 4-28  
READ/PROMPT command  
  preferable in captive command  
    procedures • 5-82  
READALL Privilege • A-11  
READALL privilege • 4-9  
Reading disks  
  to glean old information • 4-48  
RECALL/ALL command • 3-41  
Reconnection  
  disadvantage for shared account •  
    5-36  
  of process • 3-7  
  time  
    set by TTY\_TIMEOUT  
      parameter • 5-36  
Records  
  holder  
    displaying • 5-14  
Reference monitor  
  applied to network • 7-2 to 7-5

## Index

Reference monitor (cont'd.)  
  concept in security • 2-1 to 2-7  
REMOTE  
  as system identifier • 4-22  
  identifier • 5-9  
Remote  
  file access  
    how to deny • 7-8  
Remote login • 3-3  
  and system password • 5-27  
REMOVE/IDENTIFIER AUTHORIZE  
  command • 5-11  
REMOVE/PROXY command  
  and wildcard entry in NETUAF.DAT  
    • 7-22  
RENAME command  
  and file protection • 4-43  
REPLY/ENABLE=SECURITY command •  
  4-55  
Request for file access  
  how system evaluates • 4-2  
Reserved  
  identifier  
    See also Identifier, system-  
      defined  
Resource attribute • 4-35, 5-20  
  example • 5-20  
  use to change owner identifier of  
    directory • 4-39  
Restriction  
  on command usage • 5-50  
  on mode of operation • 5-49  
  work time • 5-48  
Restrictions  
  login class • 3-31  
  shift • 3-31  
Retries  
  controlling number  
    for dialups • 5-37  
Retries password • 3-32  
Rights  
  user  
    displaying • 5-14  
Rights database • 4-5, 4-17  
  creating and maintaining • 5-9, 5-14  
  display • 5-13  
Rights list • 4-23  
RMS\_FILEPROT  
  SYSGEN parameter • 5-14, 5-18

RMS\_FILEPROT SYSGEN parameter •  
  4-41

Rules  
  for specifying protection • 4-8, 4-9

---

## S

---

Scavenger  
  disk • 4-48  
Secondary  
  day  
    how to define • 5-48  
    password • 5-28  
Secondary password • 3-21  
  incompatible with network access  
    control string • 5-29  
Secure server • 3-23  
  incompatible with autobauding •  
    5-42  
  incompatible with communications  
    line use • 5-42  
Secure terminal server • 3-23  
Security  
  audit • 4-52  
  auditing • 6-6  
  breach  
    handling • 6-7  
  concepts • 2-1  
  costs of • 1-10  
  file protection  
    importance • 4-1  
  for users • 3-1 to 3-46  
  levels defined • 1-7  
  model • 2-1  
  monitoring tools  
    accounting log • 6-5  
  network  
    user considerations for • 3-34  
  password length as factor • 3-20  
  physical • 1-5  
    of networks • 7-6  
  surveillance  
    suggestions • 5-87  
  value to users • 3-46  
Security alarm  
  ACE • 4-25, 4-31  
  application example • 5-59  
  applications • 4-54

## Index

- Security attack
  - forms of • 6-1
- Security feature
  - account duration • 3-28
  - auditing • 6-6
  - breakin evasion • 3-33
  - dialup retries • 3-32
  - erase-on-delete • 5-76
  - erasure patterns • 4-49
  - highwater marking • 5-77
  - passwords • 3-9 to 3-28
  - secure server • 3-23
  - secure terminal server • 5-42
  - security alarm • 4-54
  - shift restrictions • 3-31
- Security kernel
  - defined • 2-3
- Security levels
  - determination of • 1-9
  - judging • 1-7
- Security manager
  - and cluster manager • 8-1
  - group assignment • 4-6
  - personal account • 5-2
  - privilege requirements • 5-2
- Security managers
  - goals of • 1-1
- Security operator
  - terminal • 4-31
- SECURITY Privilege • A-11
- SECURITY privilege • 5-26
  - to enable alarms • 4-31
- Security problem
  - anonymity of network and dialup users • 5-49
  - automatic login accounts
    - how to reduce • 5-46
  - images linked with traceback or debugging • 5-56
  - network protected communications • 7-6
  - telephone system as • 6-11
- Security problems
  - categories of • 1-2
  - of networks • 1-6
- Sensitive
  - information
    - and the network • 7-7
- Server
  - secure terminal • 3-23
- Service
  - denial of
    - potentially induced through evasive action • 5-41
    - severe condition • 5-41
- SET ACL command • 4-21
  - default protection
    - example • 5-19
  - example • 7-22
- SET ACL/ACL command
  - example with wildcards • 4-42
- SET ACL/ACL/DELETE command • 4-29
- SET ACL/LIKE command • 4-42
- SET ACL/OBJECT command • 4-30
- SET ACL/OBJECT=DEVICE command • 5-47
- SET AUDIT command • 4-31
  - suggested auditing applications • 6-6
  - use to detect privileges used for file access • 5-59
- SET AUDIT/ENABLE/ALARM command • 4-55
- SET DEVICE/ACL command • 4-30
- SET DIRECTORY/ACL command
  - example • 5-22
- SET DIRECTORY/OWNER command • 5-20
- SET DIRECTORY/PROTECTION command • 4-40
- SET FILE/ACL/DEFAULT command
  - example • 7-22
- SET FILE/ERASE command • 4-49
- SET FILE/OWNER command • 5-20
- SET FILE/OWNER\_UIC command
  - use to change directory owner • 4-38
- SET HOST command
  - and remote access • 5-29
- SET PASSWORD command • 3-15
  - and captive accounts • 3-13
- SET PASSWORD/GENERATE command • 3-15
  - and minimum length • 5-31
- SET PASSWORD/SECONDARY command • 3-23

## Index

- SET PASSWORD/SYSTEM command
  - use to set system password • 5-26
- SET PASSWORD/SYSTEM/GENERATE command
  - to invoke generator for system password • 5-27
- SET PROCESS/PRIVILEGES command • 5-52
- SET PROTECTION command • 4-16, 4-40, 5-19
  - changing directory protection • 4-15
- SET PROTECTION/DEFAULT command • 4-41, 5-14
- SET PROTECTION/DEVICE command • 4-30, 5-47
- SET TERMINAL/DISCONNECT • 3-7
- SET TERMINAL/DISCONNECT command • 5-36
  - role against password grabber • 5-43
- SET TERMINAL/HANGUP command • 3-45
- SET TERMINAL/NOAUTOBAUD • 3-12
- SET TERMINAL/NOMODEM/SECURE command • 5-42
- SET TERMINAL/SECURE command • 5-42
- SET TERMINAL/SYSPWD command • 5-26
- SET VOLUME/ERASE\_ON\_DELETE command • 5-76
- SET VOLUME/NOHIGHWATER command • 4-50, 5-77
- SET VOLUME/OWNER command • 5-20
- SET VOLUME/OWNER\_UIC command
  - use to change volume owner • 4-37
- SET VOLUME/PROTECTION command • 5-15
- SETPRV privilege • 5-52, A-11
- SHARE privilege • A-12
- Shared files
  - considerations for a VAXcluster • 8-3
- Sharing files
  - in network environment • 7-20
- Shift restrictions • 3-31
- SHMEM privilege • A-12
- SHOW ACL command • 4-21
- SHOW CHAR
  - NCP display • 7-18
- SHOW DEVICES/FULL command
  - use to display volume owner • 4-37
- SHOW LINKS
  - NCP display • 7-18
- SHOW PROTECTION command • 4-41
- SHOW USERS command
  - and disconnected jobs • 3-44
  - and WORLD privilege • 5-70
- SHOW/IDENTIFIER AUTHORIZE command • 5-13
- SHOW/IDENTIFIER/FULL AUTHORIZE command • 5-14
- SHOW/RIGHTS AUTHORIZE command • 5-13
- Spawning of processes
  - security implications in captive accounts • 5-79
- Stopping jobs
  - with shift restrictions • 3-31
- Strings
  - network access control • 3-34
- Subdirectory
  - inherits ACL of parent • 4-40
- Subjects
  - in security model • 2-2
  - role in security • 2-4
- Submittal job
  - affected by shift restrictions • 3-31
- Subprocess login • 3-5
- Supersensitive account
  - use of dual passwords • 3-22
- Surveillance
  - by security manager suggestions for • 5-87
- Syntax
  - ACE • 4-26 to 4-33
  - Identifier • 4-23
  - identifier • 4-21
  - protection code • 4-8
  - UIC • 4-4
- SYSS\$ANNOUNCE • 5-35

## Index

SYSSNODE • 5-36  
SYSSWELCOME • 5-36  
SYSALF.DAT • 5-43  
SYSGBL privilege • A-12  
SYSLCK privilege • A-12  
SYSNAM privilege • A-13  
SYSPRV  
    and SYSTEM category • 4-6  
SYSPRV privilege • 4-9, A-13  
    affect on ownership privilege • 4-37  
SYSTEM  
    user category • 4-6  
        access to magnetic tape • 4-9  
System files  
    auditing recommendations • 6-6  
System manager  
    group assignment • 4-6  
System password • 3-12, 5-26 to 5-28  
    as cause of login failures • 3-30  
    disadvantages • 5-28  
    guidelines • 5-27  
    incompatible with LAT-11 terminal  
        concentrator • 5-26  
    lacks minimum length requirement •  
        5-31  
    recommended change frequency •  
        5-30  
    where stored • 5-27  
System privilege • 5-53  
System programmer  
    group assignment • 4-6  
System programs  
    and ACL applications • 5-70  
System-defined identifier • 4-22, 4-23  
SYSUAF.DAT  
    and rights database  
        AUTHORIZE coordinates  
        changes • 5-9  
    effect of changes on NETUAF.DAT •  
        7-16  
    normal protection • 5-34

---

## T

Tailoring  
    DSE  
        reference • 5-76

Tampering  
    system file  
        how to detect • 6-6  
Tape  
    See Magnetic tape  
TECO Editor  
    reasons to ban from captive  
        command procedure • 5-83  
Telephone system  
    as a security problem • 6-11  
Template  
    use for account creation • 5-2  
Terminal  
    access  
        controlling through system  
            password • 5-26  
    applications  
        how to limit access • 5-47  
    hardcopy  
        logout considerations • 3-44  
    security operator • 4-31  
    system password requirement for •  
        3-12  
    usage restrictions • 5-47  
    video  
        logout considerations • 3-42  
        virtual • 3-7  
Terminal concentrator  
    See also LAT-11  
    considerations for breakin detection  
        • 5-39  
    effects on login • 3-3  
Termination  
    abrupt  
        due to change in day definition •  
            5-48  
    job  
        imposed by shift restrictions •  
            3-31  
Tied account  
    See Captive account  
Time of day  
    restrictions  
        for login • 3-31  
TMPMBX privilege • A-14  
Traceback  
    as security hazard • 5-56

## Index

Training  
  user  
    importance to security • 5-62  
Trojan Horse  
  precautions against • 5-71  
Trojan horse • 4-51, 5-55  
Trouble indications • 6-2  
TTY\_DEFCHAR2  
  SYSGEN parameter  
    enabling system passwords for  
      remote login • 5-27  
    use to disable virtual terminals •  
      5-36  
TTY\_DEFPROT  
  SYSGEN parameter • 5-47  
TTY\_OWNER  
  SYSGEN parameter • 5-47  
TTY\_TIMEOUT  
  SYSGEN parameter  
    set reconnection time • 5-36  
Turnkey account  
  See Captive account  
Turnkey application  
  use ALF to establish terminals •  
    5-45

---

## U

UAF  
  and privileges • 5-52  
  modifications  
    and security audit • 4-54  
UIC  
  alphanumeric  
    internal handling • 5-9  
  format • 4-4  
  reissue  
    caution against • 5-12  
  role in security • 2-5  
  sharing same • 4-26  
  syntax • 4-4  
  translation and storage • 4-5  
  uniqueness requirement  
    for VAXcluster • 8-3  
UIC identifier • 4-21, 4-22  
  deleted  
    how to recognize • 5-12  
UIC-based protection • 4-1 to 4-16  
  defined • 2-5

UIC-based protection (cont'd.)  
  introduction to • 4-1  
Unfounded reports  
  proper handling • 6-3  
User  
  account  
    guidelines for establishment • 5-2  
    to 5-67  
  categories • 4-1  
  introduction to system • 5-62  
  password  
    defined • 3-10  
  privilege  
    granting • 5-51  
  rights  
    displaying • 5-14  
User anonymity  
  and dialup lines • 3-32  
User category • 4-6  
  omission from protection code • 4-9  
  sequence in which checked • 4-10  
User identification code  
  See UIC  
User irresponsibility  
  as security problem • 1-2  
  training as antidote • 5-62  
User penetration  
  as security problem • 1-4  
User probing  
  as security problem • 1-3  
Username  
  as identifier • 4-22  
  revealed at logout • 3-43  
  role in security • 2-4  
Username prompt  
  use ALF to suppress • 5-45  
Usernames  
  avoid tell-tale • 5-55  
Users  
  value of security to • 3-46

---

## V

VAX/VMS Mail Utility  
  See MAIL  
VAXcluster  
  security considerations • 8-1  
Verification  
  of circuit • 7-9



## Index

Verification (cont'd.)  
  of user identity  
    mandatory • 5-28  
Video terminal  
  clearing screen • 3-43  
  logout considerations • 3-42  
Virtual terminal • 3-7  
  disabling • 5-36  
Virtual terminals  
  and logout • 3-44  
VOLPRO privilege • A-14  
Volume  
  erasures  
    for security reasons • 5-76  
  protection • 4-3, 4-14  
Volume protection  
  lack of default for • 4-40  
VTA  
  as indicator of virtual terminal • 3-44

---

## W

---

Weekday  
  restrictions for login • 3-31  
Welcome message • 3-7  
  security disadvantage • 5-36  
White collar crime • 1-5  
Wildcards  
  and AUTHORIZE proxy command •  
    7-22  
  in ACL commands • 4-42  
  in identifiers in ACEs • 4-29  
  use in ADD/IDENTIFIER command •  
    5-10  
  use in SHOW/RIGHTS command •  
    5-13  
Work  
  days  
    and restrictions • 5-48  
  hours  
    and restrictions • 5-48  
WORLD  
  user category • 4-6  
WORLD privilege • A-15  
  impact on SHOW USERS command  
    • 5-70  
Worm • 5-73  
Worm-hole • 5-73  
WRITE access • 4-7

WRITE access (cont'd.)  
  meaning for directory file • 4-12  
  meaning for disk file • 4-11  
  meaning for volume • 4-14  
  meaning in ACL • 4-28  
Write-only file  
  not supported by VAX/VMS • 4-11



## READER'S COMMENTS

**Note:** This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well organized? Please make suggestions for improvement.

---

---

---

---

---

---

Did you find errors in this manual? If so, specify the error and the page number.

---

---

---

---

---

---

Please indicate the type of user/reader that you most nearly represent:

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) \_\_\_\_\_

Name \_\_\_\_\_ Date \_\_\_\_\_

Organization \_\_\_\_\_

Street \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip Code \_\_\_\_\_

or Country

Do Not Tear - Fold Here and Tape

**digital**



No Postage  
Necessary  
if Mailed in the  
United States

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

SSG PUBLICATIONS ZK1-3/J35  
DIGITAL EQUIPMENT CORPORATION  
110 SPIT BROOK ROAD  
NASHUA, NEW HAMPSHIRE 03062-2698



Do Not Tear - Fold Here

Cut Along Dotted Line